ZX-Appeal

vancouver sinclair
users group

1987

★★★ Summer Mega-Issue ★★★

M-M-MAKE MINE
SIN-SINCLAIR...OK!

THIS ISSUE.........................

Summertime, and the living is easy!! At least I hope you're all having a great one. But just because the weather has been great don't forget your little companion that kept you company through the long hard winter...no, your computer. I hope this issue will have a little something for everyone. The Zeeper 'graces' our presence again - I know but what can I do. Harvey T. gives us the latest installment of 'Playing With Electricity' as well as the minutes of the last meeting. Rois H. emptied all the Saudi Arabian sand out of his QL and offers a neat item for you QLers - QL to PC-XT file transfers. John B. chastises man for his inability to keep up with the computer. Ken A. sent along a great in-depth look at the hardware side of the PC8300. I've read other articles about this machine but Ken's is the most comprehensive by far. We continue with the next segment of 'Buzzwords'. Being the summer Mega-Issue you're expecting some really great reprints from the NETWORK. Well, you've got them: hardware projects, software reviews and listings,, tips, hints, and more; as well as a reprint of how to turn that extra 1000 into a programmable printer buffer. Enough of this, lets get on with it.

****

BITS & PIECES.........................

...we have our first contributions to the club 2068 program library: Hugh P. sent along a GREAT action arcade-type game that he painstakingly typed in from a Brit mag; new member Joan K. sent in a tape chock o' full of hints, tips, and just plain good stuff. Many thanks to each of you. If anyone else has Public Domain stuff that others would enjoy, send it along to Ian.

...you might have heard that ZX Computing, the only Brit mag available locally, has ceased. It is an ex-mag. And it ain't just resting. Argus Specialist Limited, the publisher, says something about "the closure is due to a decline in the market place where this magazine previously operated." What did they expect after turning it into a games mag for kids!?!

...the ComputerFest was such a hit that Fests are being planned for Florida next Feb or Mar; next May in either/or/both Cleveland, and San Francisco; and we are having our own in Seattle on the 26th of September...thats next month. More info about the Seattle Fest will appear in the next issue and be given at the meeting but lets have everyone think about it between now and then. Maybe we'll car-pool or even rent a small bus or van. Think about it.

...VSUG member Bill J. of Panama City, FL., will be publishing his own quarterly newsletter 'devoted to the support of users who have up-graded to Disk Drive systems'. $12.00 per year and guaranteed to satisfy or the balance of your money back.

...Zebra Systems is still in the picture in a big way. A member received his NEW Zebra catalog - new items added and even lower prices.

...we're glad to have a new exchange partner in the NETWORK - the Dallas Timex/Sinclair/Amstrad Users Group.

...want your QL to run faster? Replace crystal X1 with one that runs at 16 Mhz. Won't run 9600 bd at the serial port but a true parallel port is unaffected. Otherwise works great.

...the club bought a copy of The Guide to Timex/Sinclair Telecommunications at the 'Fest. This 100 page book has been most generously placed in Public Domain by the authors. There isn't anything that has not been included regarding telecommunications for our machines. Copies will be made and given to those members indicating they want a copy. Hold up your hand at the Sept meeting. A count will be taken and copies made.

...Sinc-Link, the N/L of the Toronto Group reports that members have sent money to Doug Dewey but have not received their goods. Now no one can get in touch with Doug. Problems? Be advised.

...rumour has it that Amstrad is looking to bring out a 16/32 bit machine but the R&D expense is astronomical. They might just realize they already have just about the best 68000 - based machine to come out but it is 8/32 bit. Stick a decent keyboard and a couple of drives or a hard disc into a proper case with a QL motherboard and VOILA. Hey Alan want some Sugar on your crow?

...the report of the demise of the Ottawa-Hull group was a little premature. A new address came to light and we're back in touch.

...video tapes were taken at the 'Fest of all 12 seminars. The cost of acquiring all 12 tapes is only $40.00US so the club might do so. We'll talk about it at the next meeting.

****

Minutes June 12/87 (23 present)

-by your HUMBLE scribe

The meeting was opened at approximately 19:25 hours by Ken the Prez who cast imprecations in the general direction of all scribes with watches among whom your present reporter is not numbered. At any rate, we did begin & the time was not 19:00.

First, we welcomed to our ranks Hilda McKinnon a new member from Vancouver. Ken then launched into a short demo of the PC8300 which he had received from American Design Components. He played 'Oh Suzanna' at a variety of speeds for us. The machine has a range of 3 octaves. It was noticed that that the ROM is completely different from the ZX81 & will not run the same machine language.

Rod Humphreys then reported that Tim Stoddard still has the great 64K rampack deal going. The Product Profile/87 was this week at the PNE & several of the members went to pick up freebies & data sheets & such. Rod also exclaimed that Her Majesties Royal Snails have set a new land speed record of 1 day to deliver a letter from Vancouver to Burnaby. We have a new member Jeanne Kiley from the USA who has the good sense to want to zap the zeeper.

Tim Woods of Time Designs Magazine has raised the possibility to Rod of having a Pacific Northwest Sinclair Computing Conference this coming fall, specifically Sept.26/87 in Seattle. Any comments and feedback about this idea are welcomed.

Rod then put on another hat, and mentioned that we still have about $700.00 in the credit union. Printing is now being done at Rod's place of employ & there is a nominal charge of a penny a page which is keeping our printing costs down - still.

Rod then gave us about a 40 minute slide show of his trip to the Indiana Sinclair Conference. He bought oodles of stuff & had a good time meeting all the Sinclair notables he could. The slide show was excellent. He taped his talk so that he could pass it on to other groups & people. There are apparently videotapes of some of the lectures given, which we will be trying to arrange in the future.

John Brohman then spoke up & mentioned he is now selling Specterm 64, Pixel Print & the Larken Disk Interface for the 2068 via Weymil.

It was moved by Bob Denison & seconded by Harvey Taylor that Karl Brown be voted a Lifetime Honorary Membership in recognition of his uniquely valuable contribution to the club in its formative years. The motion was passed by acclamation. Those members out of town might want to check out ~~May~~/APRIL(87) Byte magazine to see what Karl has been up to lately. As a result of those articles, letters have been pouring into Al Wright's office where the Robotics club meets & Al is getting a Robotics newsletter together. Subscriptions are $15.00. There will be no Robotics meetings until Sept, by the way.

Rod mentioned at one point that our new member Jeanne Kiley had suggested that the way to overcome the sudden drop in attendance any time elections are being held is to announce that all of those not present are automatically nominated. Hmmmm...

Wilf & Harry reported, for the hardware group, that the 32K nonvolatile ram board was finally about to hit production. Wilf says he will have boards by next week. Harry Slot has designed a monster 224K nonvolatile bankswitched nonvolatile ram disk board for the ZX81. He had the unpopulated board with him. There will no doubt be articles about these projects forthcoming.

Rod rose to thank all of those who have sent in newsletter articles and to encourage the reticent to take pen in hand. [wordprocessor in x?]

Ian the Librarian stood to say that he was getting a collection of books and magazines & to keep them coming. He is specifically looking for documentation for the following programs: StarTrek, Mars, Frogger, Financial Manager, Screen Calculator, Robbers of the Lost Tombs, ChekStub & Morse Code Translator. If you can help, drop him a line c/o Rod. He is also looking for Conversational French & any Educational Programs. He is now using a dubbing deck for making copies & it seems to work fine.

Harvey presented Wilf Rigter with a copy of Fractal as an adjunct to winning his QL last month.

Mario Vereira showed us a small monitor driver and tape loader board he had made with an LED for watching levels. Guido Vereira by the way needs help getting his SINCLAIR INTERFACE 1 back up and running. If anybody has one of these beasts, will they please get in touch with Guido and see if some troubleshooting arrangements can be made.

Ken the Prez mentioned that when he sent Tim Stoddard a non working 16K pack, he got a neat letter from Tim documenting why it didn't work, as well as his 64K pack. Rod told us about getting two 2816's from Tim in return for considerations given.

Rois Harder told us that he had finally got his QL from Saudi Arabia this very day. It seemed none the worse for wear after its travels.

The next meeting will be Sept.12/87. There will be a super newsletter in August & a short reminder newsletter in early Sept.

THE SET THAT STOLE THE SHOW

THE SINCLAIR MICROVISION POCKET TV RECEIVER provided a world wide sensation when shown for the first time at the recent 1966 Radio and TV Exhibition. This fantastic British set tunes over 13 channels on bands 1 and 3. operates from six self-contained "Penlite" batteries and measures only 4in. x 2½in. x 2in. Despite the minute proportions of this 30 transistor receiver, quality from the exclusively designed tube and loudspeaker is superb. This amazing Sinclair triumph will be available early in 1967 at a cost of 49 gns.

SINCLAIR MICROVISION
The world's only pocket T.V.!

THE ZEEPER SPEAKS...

I thought I would surprise you by actually saying some nice things about you miserable wretches. After spending a week talking to my brother Zeepers, I must admit to a certain level of admiration towards you. The most common trait of the Sinclair user is that collectively you are the absolute stingiest, cheapest bunch of people in the world. That may also be your most redeeming virtue.

My fellow Zeepers are continually telling how in the world of real computers, every one is going for bigger and faster systems. In the real world, everybody is impressed by megabytes and nanoseconds. What they don't realize is that their applications will use no more than 10% of the power of these new muscle machines.

The Sinclair user has an absolute terminal allergy against spending a dollar. This has given rise to an incredible level of ingenuity. The ZX-81 has to be the worst excuse for a computer ever created. That hasn't stopped you fanatics from producing miracles from literally nothing. You have made that fly-speck computer talk, sing, run disk drives, telecommunicate, use joysticks, use bubble memories, and even multi-task. You continue to produce software that accomplishes all of the standard computer tasks plus a few more. That is an incredible accomplishment considering the fact that you're all mad.

You weirdos continue to squeeze every bit of power out of your little microchips while the rest of the world continues to waste more and more computing power. If you fanatics ever got your grubby little paws on some free IBM PC's, the world would never need another computer. You guys are still trying to find uses for 64K! Imagine what would happen if you had 640K! Fortunately that will never happen. My fellow Zeepers won't allow the price of real computers to come within the range of the ZX-81 user. They prefer that you stay contained in a quarantined population. The thought of even 100 ZX-81 users running amuck with real computers is to horrible to contemplate. The world isn't ready for really efficient software or simple hardware. They are perfectly content spending thousands of dollars to do $20.00 worth of work.

Next meeting:

SEPT/87						
SUN	MON	TUE	WED	THU	FRI	SAT
*	*	1	2	3	4	5
6	7	8	9	10	**11**	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	*	*	*

Sinclair-QL to IBM-XT File-Transfer
by Rois Harder... July 4/87

From start to finish it was a challenging project... transferring a few hundred pages of Quill documents and program files from my VER$-JM British-model QL, to my IBM-XT computer.

After waiting six months for my personal effects to arrive from Arabia, I was curious especially about how the many Microdrive Cartridges had stood up to climate and possibly worse conditions in the hands of the foreign shippers.

I did request time-to-time assistance from a few chaps on various aspects of this challenge, and Harvey Taylor was one who provided timely aid as I "slogged away" often into the wee hours of the morning. This was my first attempt at interfacing computers directly, and the matter consequently proceeded at a snail's pace.

I initially had some concerns about my 50HZ units bought in London and Jeddah, but after purchasing a Step-Up Auto-transformer for the QL system, I was soon reassured about that... everything worked very well indeed when finally connected.

My attached drawing shows the final connections for transferring, and also indicates my Miracle Systems Serial-Parallel Centronics Interface bought in Jeddah as well. This unit especially came in handy for the transfer, as it was equipped with the correct 630A BT plug, a requirement for the strange SER jacks on the rear of my QL.

The drawing is self explanitory... it was kinks in programming facets of the transfer that took the longest to solve. I have an internal modem... but could not use that for a direct connection between computers, so by inserting DB25 plugs in my Miracle System's cable, I finally configured the leads/wires correctly.

Another helpful chap from my work soon convinced me that the PROCOMM terminal software could do the IBM receiving of the ASCII files quite nicely. Assistance from another fellow provided an alteration to my QL-Super-Basic program so that the strange ASCII-CR character codes could be converted.

It was a pleasant experience finally, to see my files being transferred and displayed on both the outgoing and incoming computers' monitors. The two-minute per page rate was rather slow, but certainly more convenient than re-typing the many files into the IBM.

Finally, rather than removing other control codes from the files in the IBM, I found the "TEXTCON.EXE" utility in my possession would do this neatly and quickly. A 5-page file took only 20 seconds to strip in preparation for transferring the programs internally to my WORDSTAR directories.

Yet still, all files had to be "cleaned" and "reformed" while in document mode in WORDSTAR... all in all, a very time-consuming affair.

As I had not looked too deeply into the operation of the QL in Arabia, I was surprised now to notice during these ASCII transfers, just how much "extra" overhead the QUILL files especially have tagged on both fore and aft. After a document's last typed word, there were sometimes a few hundred extra characters (and blanks) that were transmitted before EOF. It was always welcome to finally see that strange "'1;E" combination that told me "the end was near".

If I might add, there were certain shortcomings that I had found while typing long documents with the QUILL wordprocessor. Although I found it much easier to use than any other to date, I was limited by the size of a file that could be transferred to microdrive. Possibly this was just my version that had the problem... I'm not sure. I have attached a small "warning" note that I heeded while using QUILL.

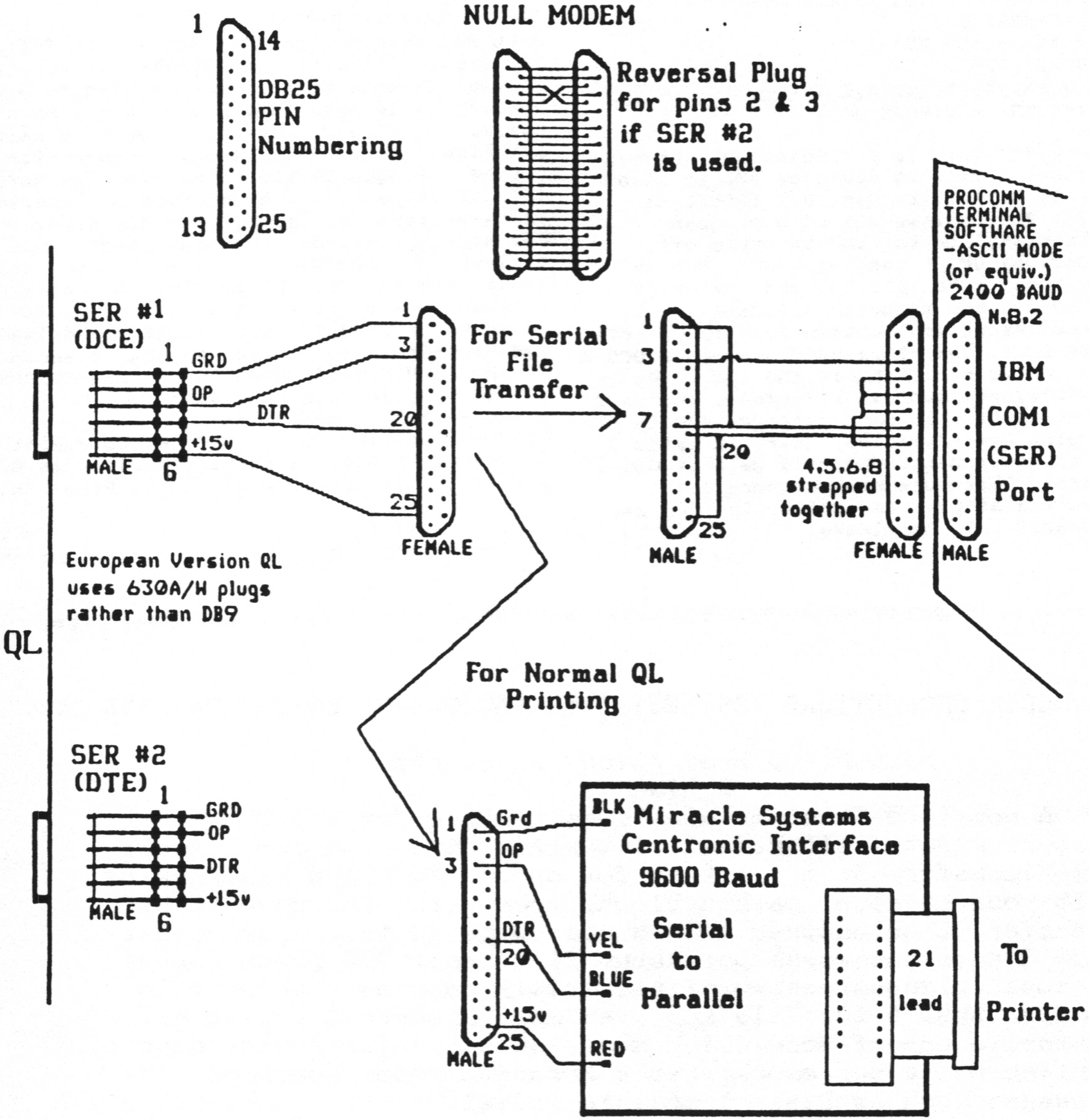
For those interested, I have the following hardware and software for my QL. (1) EPSON FX-80 Printer (2) PHOENIX color & monochrome monitor (3) MIRACLE SYSTEM'S S-P Printer Interface (4) STAG 500va 50/60 HZ Voltage Regulator(we had diesal power in the Yanbu Compound) (5) 110v-220v Step-up transf. (6) 220v Power Supply for QL. (7) 80 Microdrive Cartridges (and a few spares) (8) QL-Doctor, QL-Art, Sketchpad, GrapiQL, Air Traffic Controller... Software. (9) A few QL reference books.

I am again living at 995 Shakespeare Avenue, Tel's. 980 4167, 980 3733.

sinclair

QL

SINCLAIR-QL TO IBM-XT INTERFACING



Note: The following BASIC program will transfer files from the QL:

```
10 BAUD 2400  20 DIR MDV1_  30 LET D$="MDV1_"  40 INPUT "INPUT FILE TO SEND:";F$
50 OPEN_IN #5,D$ & F$:OPEN_NEW #6,SER1
60 IF EOF(#5) THEN CLOSE #5:CLOSE #6:PRINT:PRINT F$;" UPLOAD COMPLETE":GO TO 70
62 A$=INKEY$(#5):PRINT A$;:IF A$=CHR$(15) THEN A$=CHR$(13)
64 PRINT #6,A$;:GO TO 60  70 X%=200:Y%=820  80 BEEP 0,X%,Y%,10,10,14,1,2
90 FOR A=1 TO 1000:NEXT A     100 BEEP
```

It is important that line 62 exists. The IBM has trouble with CHR$(15)

Sketch by Rois Harder (using FONTASY)

```
10 BAUD 2400
20 DIR MDV1_
30 LET D$="MDV1_"
40 INPUT "Input ASCII file to send:";F$
50 OPEN_IN #5,D$ & F$:OPEN_NEW #6,SER1
60 IF EOF(#5) THEN CLOSE #5:CLOSE #6:PRINT:
   PRINT F$;" Upload completed":GO TO 70
62 A$=INKEY$(#5):PRINT A$;:IF A$=CHR$(15)
   THEN A$=CHR$(13)
64 PRINT #6,A$;:GO TO 60
70 x%=200:y%=820
80 BEEP 0,x%,y%,10,10,14,1,2
90 FOR a=1 TO 1000:NEXT a
100 BEEP
110 REMark *** This is a Sinclair-QL program
   to upload a file to SER#1 or #2. It also
   converts Hex(OF) to Hex(OD)..(ASCII 15
   to 13). This allows use of a program
   "textcon.exe" on the IBM to strip off
   all unusual QUILL control codes such as
   "underline" & "bold". The program calls
   the MDV-DIR and requests filename input.
   The hardware communication link is QL-SER1
   to IBM-COM1. The "terminal" program PROCOMM
   was used for receiving in the IBM...(Pg-Dn
   key,ASCII,and Download Filename.) The BAUD
   rate seems to make no difference due to the
   stripping action. In any case, it takes 2
   minutes to transfer one full page of QUILL
   document. There are many strange QUILL
   characters at page-end before the EOF causes
   the transfer to terminate.
```

****** WARNING ******

A QL Microdrive Cartridge has around 208-223 Sectors of 512-Byte Data Blocks.(110-Kbytes) It has 14-Byte Sector Headers, 2-byte Block Headers, & 10-Byte(Chr$)Medium Names.Each of maximum 42-File Headers, (5 Sectors reserved per File), is 64 Bytes long. Maximum File or Document length is approximately 5600 words. (32,768 Bytes at 5.8 bytes/Word or a maximum of 64-Sectors x 512-Bytes = 32,768 Bytes.) For safety, maximum document length must be limited to 5500 words. Files can be merged later into one for out-putting to a printer. (Allow an average of 85 words per sector.) (There is a 23,226 word limit per drive) One Microdrive loop takes approx. 7 seconds.

DO NOT TOUCH TAPE SURFACES FOR ANY REASON.

Sinclair QL - VER$ JM

Program Cartridge Copy 1.Power Up 2.Press F1 3. Master in mdv2_ 4.Formatted Blank in mdv1_ 5. Then Type:lrun mdv2_clone 6. Press Enter.

mdv1_ mdv2_

****

0 REM

John Brohman

I wonder how much longer we will have to wait for the rest of the computer revolution? Technology has successfully delivered the computer to the point where it is easily obtainable by the average working person. More and more computing power is being squeezed into smaller and smaller packages. In terms of power for the dollar, today's computers are true bargains. That was the first part of the revolution.

The next stage of the revolution will have to wait for man. I am afraid that we now have computers that we are not smart enough to fully utilize. Consider how we communicate with computers. The human brain can process many levels of information instantly. The computer has the same capability. When the brain talks to the computer, it is seriously handicapped by the keyboard. We are spending too much time typing and not enough time thinking. Technology has allowed the computer to recognize and interpret human speech for several years. It is even within the capability of the TS 1000. Computers have been able to talk to us for years. Why are we still using keyboards?

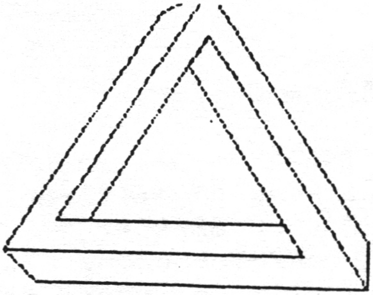
Every computer requires that we learn it's language. This is the perhaps the most serious flaw in computer design. A tool by definition, must work for man. Man does not work for the tool. Computers are designed in direct violation of the very basic concept. I don't want to learn the operating parameters of a computer. The role of the computer is to learn MY operating parameters. A computer should learn from me. I don't wish to learn from it.

Every time you turn on a computer it is just as stupid as the day it was assembled. Computers should be designed to accumulate and distinguish sets of instructions the user gives it. Once again, these instructions should be within my parameters, not the computer's. Imagine teaching a computer by voice one time to store someone's phone number and dial it when requested. Imagine never having to teach the computer how to do this again. We have the technology to do this. We just haven't learned how to do it simply.

The concept of Arthur C. Clarke's HAL is well within the capability of the average computer. I should be able to come home and have my computer detect and identify my presence. It should automatically unlock the door. It should greet me upon entering. It should have automatically recorded such things as news reports, television shows, and phone calls. It should have the temperature of the home regulated for occupancy as opposed to vacancy. It should have the hot water heater ready for a shower. It should advise of any visitors during my absence. It should advise me of any scheduled events. It could advise me of energy usage and costs. We can build houses that do this and more. We can build them marginally more expensive than existing housing. We haven't learned how. We are told we live in an information explosion. We are supposed to have a wealth of information at our fingertips. I would like to see a computer that could search both it's own database and remote ones automatically and produce the information required. Presently to do the above we have to first learn how to use a database then how to use a communications package. If we have mastered those steps; we can get retrieve the information we need. This is not exactly an explosion.

Finally, computers should stop frightening people. What is the use of having all of this wonderful technology if people are afraid to use it? The average North American sedan is very easy to drive. You get in, fasten the seat belt, start the engine, put it in gear and go. This is a far cry from the Model A with it's spark advance and crank. We are still designing computers like the Model A.

Am I asking too much? Not when you consider our technological machine can put men on the moon, develop an artificial heart, and do gene splicing.



```
  5 BORDER 0: PAPER 0
 10 CLS
 15 LET C=0
 25 FOR X=1 TO 7
 26 IF C=0 THEN LET X=7
 27 INK X
 29 LET C=1
 30 PLOT 80,50
 40 READ A,B
 45 IF A=.1 THEN GO TO 140
 50 DRAW 3*A,3*B
 60 GO TO 40
 70 DATA 18,24,-3,-4,18,-24,-3,
4,-35,0,23,30,25,-32,0,-6,-25,32
,-32,-51,0,-5,4,46,0,-46,0,24,31
,7,3,.1,.1
140 RESTORE
150 GO TO 25
```

8888888888888888

Military micro blazes along at 40 MHz

Performance Semiconductor PACE1750A VHSIC single chip, 16-bit, 40-MHz processor with on-chip 32-bit and 48-bit floating-point arithmetic implements MIL-STD-1750A Instruction Set Architecture (ISA).

It is supported by a family of advanced CMOS products used to design closely coupled computer systems.

It provides 13 addressing modes, including direct, indirect, indexed, based, based indexed and immediate long and short, and can access 4 Mbytes of memory in 128-Kbyte segments.

Throughput is 2.5 MIPS for a standard real-time integer/floating-point instruction mix.

The chip uses a single multiplexed 16-bit parallel bus. Basic bus cycle is 4 clocks long.

It packs 200,000 transistors in 52,000 square mils, consumes less than 1 watt at 40 MHz and operates over the full military temperature range of −55 to +125C.

THE SMALLEST SET IN THE WORLD

SINCLAIR MICRO-6
2 3 4 5
BRITISH MADE
BRYANT & MAY

ACTUAL SIZE

SINCLAIR MICRO-6
SIX-STAGE RECEIVER

Over 8,000 built and in use all over the world

THE SINCLAIR MICRO-6 continues unchallenged as the most remarkable receiver of its kind ever made available to the public anywhere in the world. It has special 6-stage circuitry and is, at the same time, the smallest set on earth. Everything except the lightweight earpiece is contained in the smart, minute white, gold and black case which is appreciably smaller than a matchbox, as the illustration shows. With vernier-type tuning control, bandspread over the higher frequency end of the medium waveband and powerful A.G.C. to ensure fade-free reception of the most distant stations, the Micro-6 provides remarkable standards of performance. Quality of reproduction is outstandingly good and, again and again, the set is reported to give excellent results where other sets cannot be used at all. The Micro-6 cannot be too highly recommended, both as an intriguing design to build, and a most practical radio to use.

CHASSIS VIEW—
ACTUAL SIZE

$1\frac{3}{8}$" × $1\frac{3}{16}$" × $\frac{1}{2}$".

●

Weighs under 1oz.

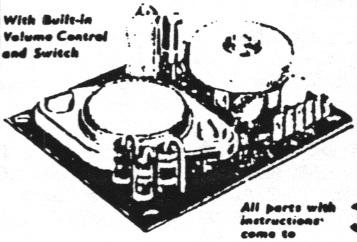
●

Tunes over M.W.

●

Plays in cars, trains, buses, modern buildings.

SINCLAIR TR750 POWER AMPLIFIER



With Built-in Volume Control and Switch

Designed specially for use with the Sinclair Micro-6

THE TR750 (for building yourself or available ready built) measures only 2in. x 2in. It will provide powerful loudspeaker reproduction from the Micro-6 which can then be used as a car-radio, or domestic or portable loudspeaker set. The TR750 also has many other applications such as record reproducer, intercom or baby alarm. An output of 750 milliwatts for feeding into a standard 25-30Ω loudspeaker requires only a 10mV input into 2KΩ. Frequency response 30-20,000 c/s ± 1dB. Power required 9 to 12 volts.

All parts with instructions come to **39/6** **READY BUILT AND TESTED** *with instructions* **45/-**

Easily built in a single evening

Using components never before made available to the public, the Micro-6 is nevertheless easy to build. All parts including lightweight earpiece and 8 page instructions manual come to

59/6

"TRANSRISTA" *black nylon strap for wearing the Micro-6 like a wrist-watch 7/6*

MALLORY MERCURY CELL ZM312
(2 required), each 1/11

Handy pack of 6 cells, 10/6

sinclair

SINCLAIR RADIONICS LTD
COMBERTON, CAMBRIDGE
Telephone: COMBERTON 682

UNIQUE GUARANTEE

The following unconditional guarantee applies to everything you buy from Sinclair Radionics Ltd.:—

If you are not completely satisfied with your purchase (we are confident you will be delighted) your full purchase price will be refunded instantly and without question.

- **FULL SERVICE FACILITIES AVAILABLE TO ALL SINCLAIR CUSTOMERS**

13WW—147 FOR FURTHER DETAILS.

PC8300 PRELIMINARY TECHNICAL REPORT

By Ken Abramson, VSUG

When Rod Humphreys brought his newly acquired PC8300 to the March VSUG meeting, rumors flew and interest was keen. Could it be that $29.95U.S. could buy a TS1000 compatible computer with a nice chicklet keyboard and a programmable sound generator? Well ... almost! While compatibility is a moot point, there is little doubt that this computer CAN be serioulsy considered as a replacement for your aging ZX81 or TS1000, but do not expect 100% hardware or software compatibility. Also, do not expect any support beyond system hardware from the supplier. AMERICAN DESIGN CO has answered my request for software and additional information with a polite, "Not available." This places the ball right back into our court. If we wish to add the PC8300 to our dinosaur collection, we had better come up with the required info and software ourselves. To this end, Harry Slot and I have made some preliminary observations that might help people decide whether they should be interested in the PC8300, and will also indicate areas for caution.

Manuals: Two PC8300 instruction manuals come with the computer: an English version (50 small pages) and a Chinese version (86 large pages).

The English manual is a concise little booklet (full of grammatical mistakes) giving operating instructions and simple BASIC programming lessons. Its main usefulness to the average user who knows BASIC will probably be Chapter 9, which outlines how to program the music and sound. Also useful is Appendix B, giving the error codes.

The Chinese manual is much more comprehensive, being a close copy of the British Sinclair ZX81 manual. It is so close a copy, however, that some ZX81 information has NOT been replaced by PC8300 information! E.g.: a 9 Volt power supply is shown, whereas the PC8300 uses a 12V power supply. This is somewhat dangerous because one assumes that the edge connector diagram given on page 66 shows the correct voltages and bus connections for PC8300 interfacing. THIS IS NOT THE CASE!!! What IS shown is an exact copy of the ZX81 edge connector diagram! A voltmeter will show that instead of the 9 Volts shown in the manual, about 14 Volts is present (12 Volts unregulated from the 12 Volt power supply)!! Do NOT use a TS1016 RAMpack with the PC8300!!! Other PC8300 edge connector differences are described below.

Suffice it to say, the Chinese manual is a close copy of the British ZX81 manual, right down to the programming examples used (e.g.: A$="DOUBLE GLOUCESTER"). The majority of new info is again given in the chapters on Sound and Music, Memory Organization, System Variables and the Appendix showing the Character Set.

HARDWARE: Two brand names have been reported for the 8300: LAMBDA and UNISONIC. Also there may be slightly different production versions of the motherboard.

The computer consists of five chips, four transistors (excluding two inside the RF modulator), a five volt regulator, and an assortment of the usual diodes, resistors, capacitors. A 1.5 inch loudspeaker is also seen. The chips consist of a 2016 2K RAM, an 8442 ROM, a D780C (Z80A CPU), a 74LS05 and a C4005 ULA/ I/O chip. All chips are socketed except the 74LS05. There are four sets of empty chip pads for 2114 RAM chips, probably used as an alternative to the 2016 RAM.

The PC board is mounted to the unshielded cream-colored plastic case by four small mounting screws. CAUTION: the plastic case has been found to generate significant levels of static electricity!

The neat rubber 'chicklet' keyboard is connected to the motherboard by means of a very stiff sixteen conductor transparent ribbon cable soldered at both ends. Harry and I have replaced this in both our computers by a flexible ribbon cable and a plug and socket arrangement in order to avoid cable damage when the computer is frequently opened up for examination or modification.

The rear of the computer reveals the following sockets: DC POWER, EAR, MIC, MONITOR, EXTERNAL BUS (edge connector), JOYSTICK, and TV. The MONITOR output jack is driven by a transistor, and should therefore be capable of running a high or low impedance monitor. There is NO volume control for the sound (Harry and I have both installed one without difficulty).

The video signal appears a little distorted on a TV or a monitor, since the tops of some characters tend to lean a bit. This might be caused by excess video or by contention between the sync pulses and the black background levels. The joystick port accepts a standard Atari-type joystick which was found to activate a curious combination of keys: UP=4, DOWN=R, LEFT=7, RIGHT=F, and SHOOT=U. Can anybody explain the reasoning behind these choices?

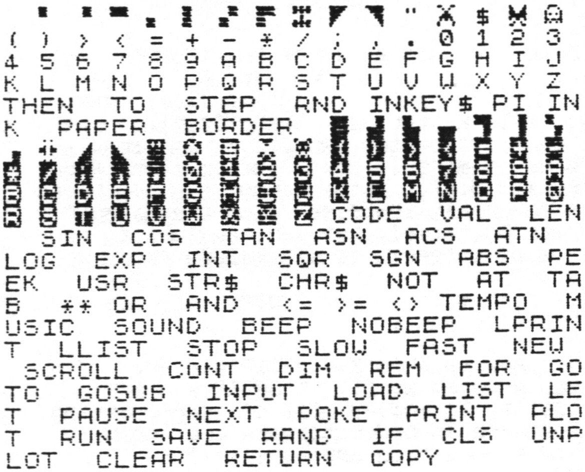
The sounds that come from the loudspeaker are of three kinds: beeps for every key except SHIFT, programmable musical notes (single voice) having a range of three octaves, and programmable games sounds. All sounds appear to be generated in the so-far mysterious C4005 ULA chip and are amplified by a transistor which drives the loudspeaker.

Last, but by no means least, is the sad saga of the PC8300 edge connector. Yes, you CAN run the TS2040 printer from it. NO, you CANNOT run all TS1000 peripherals from it, and you may even cause some damage if you try!!! As previously mentioned, 14 Volts appears where there should only be 9 Volts. This may cause a TS1016 RAMpack to blow. When I tried out my speech board (the 9 Volt line is not used by the speech board), complete silence was the result! Further investigation by Harry led to the discovery that EIGHT control lines from the CPU were not connected to the edge connector as they should have been!!! These lines were completely missing: INT, NMI, HALT, BUSAK, WAIT, BUSRQ, RESET, and M1. Since the RAMpack did not appear to use these connections, it seemed OK to me to connect them as found in the ZX81 — the speech board then worked perfectly. If you are planning to do any interfacing to the PC8300, remember: if the 14 Volts doesn't get you, the missing CPU control lines might! We are seriously considering reducing the 14 Volts to the usual 9 Volts at the edge connector and modifying the PC8300 RAMpack to operate on 9 Volts so that the PC8300 edge connector becomes truly hardware compatible with ALL TS1000 peripherals.

PROGRAMMING: It would not be fair to the developers (ZX81 cloners?) of the PC8300 if one were to describe PC8300 BASIC as a copy of ZX81 BASIC. Although ZX81 BASIC APPEARS to be about 98% compatible, there are some real differences. From the programming point of view, the main conclusion is that the majority of ZX81 and TS1000 BASIC programs can be made functionally operable with very few programming modifications. TS1000 programs can be loaded into the PC8300, but any variables saved by the TS1000 will NOT be carried over to the PC8300. If the variables were not initialized inside the program, you will have to type them into the PC8300 after you have loaded the program. This can be somewhat tedious in programs such as CONVERSATIONAL SPANISH, which contains a lot of hidden data originally saved in string variables. By the way, you cannot load a PC8300 program into a TS1000 (unless you are sneaky and make the transfer via NON-VOLATILE RAM or some such non-tape device).

As you begin to program with the PC8300, you quickly realize that the ROM is a completely different beast from the TS1000 ROM. The first signs of these differences are the different graphics characters, the missing colon, question mark and quote image, and the lack of single key token entry.

PC8300 CHARACTER SET



```
( ) > < = + - * / ; , . 0 1 2 3
4 5 6 7 8 9 A B C D E F G H I J
K L M N O P Q R S T U V W X Y Z
THEN  TO  STEP  RND  INKEY$ PI IN
K  PAPER  BORDER
                    CODE  VAL  LEN
 SIN  COS  TAN  ASN  ACS  ATN
LOG  EXP  INT  SQR  SGN  ABS  PE
EK  USR  STR$  CHR$  NOT  AT  TA
B  ** OR  AND  <= >= <> TEMPO  M
USIC  SOUND  BEEP  NOBEEP  LPRIN
T  LLIST  STOP  SLOW  FAST  NEW
 SCROLL  CONT  DIM  REM  FOR  GO
TO  GOSUB  INPUT  LOAD  LIST  LE
T  PAUSE  NEXT  POKE  PRINT  PLO
T  RUN  SAVE  RAND  IF  CLS  UNP
LOT  CLEAR  RETURN  COPY
```

There are no shading graphics characters and the screen display is inverse video (white on black background). There is an automatic line numbering key (increments by 10), color commands for use with a color board: INK, PAPER, BORDER, and sound commands like MUSIC, TEMPO, SOUND, BEEP, NOBEEP. Music data is held in strings and melodies can be assembled by concatenating the strings. Variables cannot take the same name as a token; e.g.: LET OR=5 is illegal because OR is a token. Variables and strings CAN be assigned without the use of the LET statement (e.g.: A$="SINCLAIR")! The REM statement will not hold spaces unless you use quote marks (even one set of quote marks at the beginning of the REM statement will work)! Error codes are two letter abbreviations of error descriptions: e.g.: UV=undefined variable ... what does BS mean? Why 'Bad Subscript', of course!

```
   1 REM " PC8300 MUSIC EXAMPLE
          OH SUSANNA

  10 LET A$="C2D2E4G4G6A2G4E4C6D
2E4E4D4"
  20 LET B$="C4D12"
  30 LET C$="D4C16"
  40 LET D$="F8F8A4A8A4G4G4E4C4D
12"
  50 MUSIC A$+B$+A$+C$+D$+A$+C$
```

The BASIC program area begins at address 17302 rather than 16509 used by the TS1000. This means that the address for storing the first byte of machine code in the first program REM statement will be 17307 (not 16514). The reason for this is that the PC8300 has incorporated a fixed screen memory (memory mapped display file) located between 16510 and 17235. Isn't that nice!

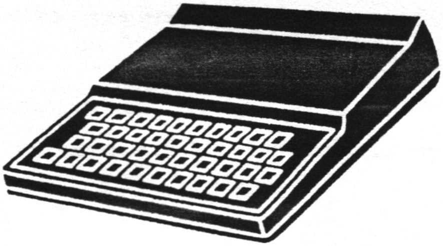
These are just a few of the PC8300 ROM idiosyncrasies uncovered so far. Harry Slot has the circuit diagrams well under way. Wilf Rigter has been looking into the workings of the ROM. I am looking forward to more in-depth information from Harry and Wilf. Both Harry and I have been favorably impressed with the hardware and its potential, but admittedly it may not be of interest to all TS1000 fans.

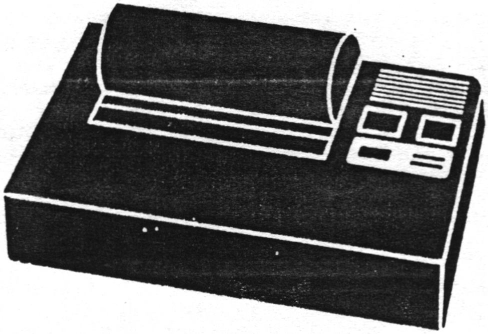
In summation, the PC8300 may be of interest to you if:

1. you are an avid ZX81/TS1000 user.
2. you program mostly in BASIC and are willing to modify most of your existing software in order to make the graphics and punctuation appropriate.
3. you do not intend to use any interfaces other than the TS2040 printer and the PC8300 RAMpacks (or you are willing to perform the required edge connector modifications in order to utilize existing ZX81 peripherals.
4. you wish to make use of the simple BASIC sound programming procedures.
5. you can move existing ZX81 machine code to the new addresses required by the PC8300 and you can replace all ZX81 ROMcalls with full machine code routines (this is one way to learn how to manipulate machine code).

The PC8300 is a versatile little computer that meets the first criterion for belonging in our club ... it's CHEAP! Keep in mind, however, the complete lack of support that comes with it (but you're already used to that, aren't you?).

****





****

```
   1 REM THREADS...by David King
 from ZX Computing June /87
   2 REM this program helps desi
gn those thread designs created
on wood with pins and thread.
  10 LET m=5.5: LET j=44/49: LET
 x$="j*d*COS z+127": LET y$="d*S
IN z+87": PAPER 0: INK 7: BRIGHT
 1: BORDER 0: CLS : GO TO 90
  20 LET z=p
  30 LET x=VAL x$: LET y=VAL y$:
 RETURN
  40 LET z=p+s*k: GO SUB 30
  50 LET u=x-PEEK 23677: LET v=y
-PEEK 23678: IF NOT q THEN LET w
=u/j: LET t=t+SQR (ABS (w*w)+ABS
 (v*v))
  60 RETURN
  70 INPUT "start new board? (y/
n) "; LINE s$: IF s$="y" THEN CL
S
  80 INPUT "use same pins? (y/n)
"; LINE p$: IF p$="y" THEN GO TO
 100+(s$="n")
  90 INPUT "width?(m for max 5.5
 inches) ";d: LET d=d*87/m: INPU
T "how many pins? ";n: LET s=2*P
I/n
 100 FOR p=0 TO 1.99*PI STEP s:
GO SUB 20: PLOT x,y: BEEP .004,0
: NEXT p
 110 INPUT "how many pins on eac
h time? ";k: INPUT "continuous o
r return? (c/r "; LINE c$: IF c$
="r" THEN INPUT "draw return thr
ead? (y/n) "; LINE r$
 120 LET z=0: LET t=z: GO SUB 30
: PLOT x,y
 130 FOR q=0 TO 2*PI STEP s: GO
SUB 40: DRAW u,v: IF c$="c" THEN
 LET p=z: GO TO 160
 140 LET p=p+s: GO SUB 20: GO SU
B 50: IF r$="y" THEN DRAW u,v: G
O TO 160
 150 PLOT x,y
 160 NEXT q: PRINT #0;"thread us
ed ";INT (m/1.75*t*n)/100;" ins"
;TAB 23;"press key": PAUSE 0: GO
 TO 70
```

BUZZ WORDS

We continue with a couple of letters from the BUZZWORD dictionary.

bBBbB

BAUD RATE: A measure of data flow. The number of signal elements per second based on the duration of the shortest element. When each element carries one bit, the Baud rate is numerically equal to bits per second (bps). The Baud rates on UART data sheets are interchangeable with bps.

BCD (Binary Coded Decimal): Each decimal digit is binary coded into 4-bit words. The decimal number 11 would become 0001 0001 in BCD. Also known as the 8421 code.

BENCHMARK: Originally a surveyor's mark used as a reference point in surveys. In connection with microprocessors, the benchmark is a frequently used routine or program selected for the purpose of comparing different makes of microprocessors. A flow chart in assembly language is written out for each microprocessor and the execution of the benchmark by each unit is evaluated on paper. It is not necessary to use hardware to measure capability by benchmark.

BIDIRECTIONAL: A term applied to a port or bus line that can be used to transfer data in either direction.

BINARY: A system of numbers using 2 as a base in contrast to the decimal system which uses 10 as a base. The binary system requires only two symbols, 0 and 1. Two is expressed in binary by the number 10 (read one, zero). Each digit after the initial 1 is multiplied by the base 2. Hence the following table expresses the first ten numbers in decimal and binary:

Decimal	Binary	Decimal	Binary
0	0	5	101
1	1	6	110
2	10	7	111
3	11	8	1000
4	100	9	1001

BRANCH: Refers to the capability of a microprocessor to modify the function or program sequence. Such modification depends on the actual content of the data being processed at any given instant.

BREAKPOINT: A program point indicated by a breakpoint flag which invites interruption to give the user the opportunity to check his program before continuing to its completion.

BUFFER: A circuit inserted between other circuit elements to prevent interactions, to match impedances, to supply additional drive capability, or to delay rate of information flow. Buffers may be inverting or non-inverting.

BUS DRIVER: An integrated circuit which is added to the data bus system to facilitate proper drive to the CPU when several memories are tied to the data bus line. These are necessary because of capacitive loading which slows down the data rate and prevents proper time sequencing of microprocessor operation.

BUS SYSTEM: A network of paths inside the microcomputer which facilitate data flow. The important busses in a microprocessor are identified as Data Bus, Address Bus, and Control Bus.

BYTE: Indicates a pre-determined number of eight consecutive bits treated as an entity. "Word" and "Byte" are used interchangeably.

CCCCC

CLOCK: A generator of pulses which controls the timing of switching circuits in a microprocessor. Clock frequency is not the only criterion of data manipulation speed. Hardware architecture and programming skill are more important. Clocks are a requisite for most microprocessors and multiple phased clocks are common in MOS processors.

COMBINATIONAL LOGIC: A circuit arrangement in which the output state is determined by the present state of the input. Also called Combinatorial logic. (See also Sequential Logic.)

COMPILERS: Compilers translate higher-level languages into machine code.

CONDITION CODE: Refers to a limited group of program conditions such as carry, borrow, overflow, etc., which are pertinent to the execution of instructions. The codes are contained in a Condition Codes Register. (See also Status Word Register.)

CONTROL BLOCK: This is the circuitry which performs the control functions of the CPU. It is responsible for decoding microprogrammed instructions, and then generating the internal control signals that perform the operations requested.

CONTROL BUS: Conveys a mixture of signals which regulate system operation. These "traffic" signals are commands which may also originate in peripherals for transfer to the CPU or the reverse.

CONTROL PROGRAM: The Control Program is a sequence of instructions that will guide the CPU through the various operations it must perform. This program is stored permanently in ROM memory where it can be accessed by the CPU during operations.

CPU (Central Processing Unit): The heart of any computer system. Basically the CPU is made up of storage elements called registers, computational circuits in the ALU, the Control Block, and I/O. As soon as LSI technology was able to build a CPU on an IC chip, the microprocessor became a reality. The one-chip microprocessors have limited storage space, so memory implementation is added in modular fashion. Most current microprocessors consist of a set of chips, one or two of which form the CPU.

CROM (Control Read Only Memory): This is a major component in the control block of some microprocessors. It is a ROM which has been microprogrammed to decode control logic.

CROSS-ASSEMBLER: When the program is assembled by the same computer that it will run on, the program that performs the assembly is referred to as the resident or native or self-assembler. If the program is assembled by some other computer, the process is referred to as cross-assembly.

CYCLE STEALING: A technique used to transfer data between microprocessor memory and external bulk memory. Data transfer is accomplished without stopping MPU operation by using microcycle time between data fetch and instruction fetch time in an instruction cycle. Creates a virtually infinite memory storage for MPUs. Applicable for 3-state output memories and MPUs. (See also DMA.)

THE FOLLOWING ARTICLE IS REPRINTED FROM THE JANUARY/87 ISSUE OF TIMELINEZ - THE JOINT NEWSLETTER OF THE THREE T/S USER GROUPS IN THE SAN FRANCISCO BAY AREA.

TS2040 Printer Switch...... by John Ezike

DISCLAIMER: Neither VSUG, ZXappeal, nor the author accept any responsibility for any damage that might occur if you attempt this project.

****

The reason for adding a switch is that even after pressing the 'off' button to turn the printer off, power is still being supplied to the printer! The on/off switch only controls the print driver. Leaving the printer on for very long periods of time could cause it to overheat and be damaged. You could unplug the power when not using the printer, however frequent plugging and unplugging can wear out the plug and jack. The idea is quite straightforward and simply involves cutting the trace through which power is supplied to the printer and jumpering the cut trace with a switch. A LED (light emitting diode) is used to indicate power is on. For this article I will use Radio Shack parts numbers as it will make them easier to locate.

PARTS LIST:

1 - SPDT MICROMINI SWITCH, *275-625
1 - LED *276-041
1 - 470 OHM RESISTOR 1/4W 5% *271-1317
22 GAUGE HOOK-UP WIRE

You can get the parts from any electronics store, just make sure the specifications are the same. Cut two 8 inch lengths of the hook-up wire and remove 1/4 inch of the insulation from all four ends. Clear your work area, making sure there are no small particles lying around that could enter the printer mechanism and damage it. Plug in the printer. Remove the paper from the printer by cutting the paper at the roll and pressing the 'on' button until the remaining paper comes out. Unplug the printer and turn it over. Remove the four screws, placing aside carefully for safe-keeping. Holding the case together, turn the printer back upright. Carefully lift off the top half of the case and place to one side. You are now looking at the component side of the printed circuit board (PCB). See Fig.1. In the top right hand corner is seen the heat sink and in front will be the printer mechanism. Two amber-colored flexible ribbon connectors are plugged into two sockets labeled J1 and J2. Carefully remove each connector by holding the sides and pulling upwards. You might have to move it from side to side while pulling. With the connectors free, remove the printer mechanism. There are two metal plates on either side bent out to form four flanges, with rubber grommets in the center of each flange. These grommets fit around four plastic posts; labeled E, F, G, & H; which are attached to the lower half of the case. Hold the PCB down with one hand and gently lift off the side of the mechanism that fits around posts G & H. Hold only the flanges while lifting. Do the same on the other side. DO NOT TOUCH THE ROLLER OR ANY OTHER PART OF THE PRINTER SINCE IT

IS DELICATE AND COULD BE DAMAGED. Place the mechanism in the top cover along with the screws and paper and place aside. Remove the PCB from the case by sliding a small screwdriver between the PCB and case and lifting. Centered on the rear edge of the PCB is the power jack. 1/4 inch below it are two groups of three feedthrough holes, each group in the shape of a triangle. Locate the feedthrough hole, labeled 'C' in Fig.1, in the group on the right. One inch below this hole is another hole, labeled 'D' in Fig.1. The dotted line joining points C and D refers to the trace connecting them on the other side of the PCB. The jagged lines crossing the trace denotes that the trace should be cut. Cut it. In the lower left hand of the PCB is a feedthrough hole labeled 'B' in Fig.1. Turn the PCB over and with a sharp knife or blade gently scrape off the protective coating to expose a 1/4 inch area of copper around 'B'. This will be the 'ground' to which you will solder one end of the LED.

Look at the LED. You should see a flat side around its rim and the shorter of its two leads should be next to this flat side. This is the cathode, or negative, lead. On some LED's you will see a notch instead of a flat side above the cathode. From the component side, push the cathode, or shorter leg, through hole 'B' and solder it to the copper area you just exposed. Make sure the solder connection is shiny and smooth. Now solder the other LED lead to either end of the resistor and solder the other end of the resistor to point 'A'. This is the positive side of the LED and the resistor acts as a current limiter and protects the LED from excessive current. Point 'A' is the lead on the disc capacitor C11, which is connected to the positive lead on the electrolytic capacitor C10. Turn the PCB over to verify this. If C11 is placed too close to the surface of the PCB, turn the PCB over and solder the resistor to the solder pad corresponding to point 'A'. We are now going to connect the switch. From the component side, push the stripped end of one of the hook-up wires through hole 'C' and solder it. Push one end of the other wire into hole 'D' and solder it. Make sure the trace between points 'C' and 'D' is completely cut. A cut about 1/10th of an inch should do it.

Now refer to Fig.2. Retrieve the top half of the case and with a drill or some other suitable device, drill a 1/4 inch hole to mount the switch. Place the two case halves together and drill another hole, just large enough to fit the LED, centered where the cases halves join. DON'T DRILL AROUND THE PRINT MECHANISM!! Mount the switch in the 1/4 inch hole. It'll be a little cramped so take your time. Solder the free end of the wire connected to 'C' to the center terminal of the switch. Solder the free end of the wire connected to 'D' to either of the two outside terminals on the switch. Replace the PCB in the lower half of the printer case, making sure to align the notches on the side of the PCB with the tabs in the case. Replace the printer mechanism. The rubber grommets in the flanges should be flush with the top of the screw posts. Plug in the two ribbon connectors, pushing down firmly but carefully to make sure they are in all the way. Bend the LED forward so it rests on the groove formed by half the hole in the bottom half of the case and protrudes about 1/8 inch outside the case. Make sure the LED leads are apart. Hold both case halves together, turn over, and replace the screws.

Plug in the power. If the LED is lighted, the switch is in the 'on' position. If it isn't, try the other switch position. If the LED still does not light, quickly unplug the power, open up the printer and check the connections, especially the installation of the LED, looking for solder bridges and/or bad joints. If no problems and the LED is lighted, insert the paper and perform the self-test procedure as described in the printer manual.

If all goes well, CONGRATULATIONS!!!

To use the printer after plugging in: turn the new switch 'on', and press the 'on' button. When finished just turn 'off' the new switch - no need to unplug. This project should also work for the Alphacom printer

Editor's Note: I substituted a Radio Shack #275-1565 push-on push-off switch for the one in the article. The placement is very important. I suggest if you do use this switch that you drill the required hole on the top surface to the left of the paper roll at least two inches from the front edge.

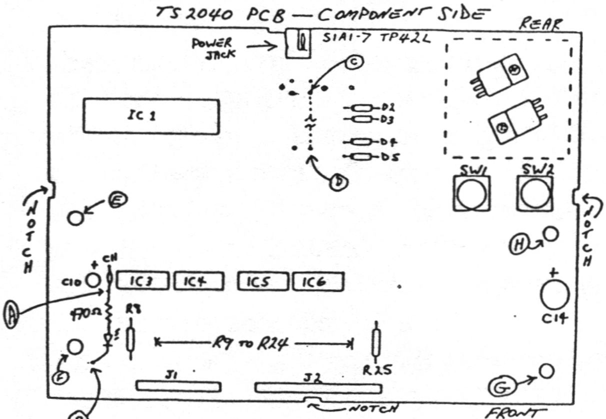


FIG. 1

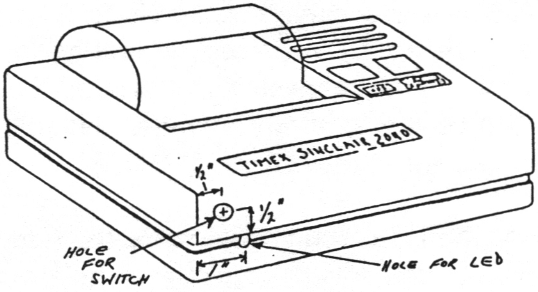


FIG 2

****

```
   1 REM by Tom Vandepoel. Repri
nted from ZX Computing May/87.
   2 REM prints to 2040 printer
in various sizes. Width cannot
be  greater than 4.
  10 INPUT "text to be printed "
;a$: INPUT "height/point ";h: IN
PUT "width/point ";w
  15 FOR n=1 TO LEN a$: LET h$="
": LET s$="": PRINT AT 21,0;a$(n
)
  17 FOR e=1 TO h: LET h$=h$+a$(
n): LET s$=s$+" ": NEXT e
  20 FOR x=0 TO 7: LET p$="": FO
R y=0 TO 7
  30 LET p$=p$+(h$ AND POINT (x,
y)=1)+(s$ AND POINT (x,y)=0)
  40 NEXT y: FOR e=1 TO w: LPRIN
T p$: NEXT e: NEXT x: NEXT n
```

Playing with Electricity -Summer/87

by Harvey Taylor

This is the second time I have started this article. The first version is in limbo lost in a mess of disks somewhere & it will be quicker to rewrite than to search through 20 disks & half a dozen corrupted remnants. I find Quill is great at trashing disks. It's usual trick is to write a file into the directory.

Anyway, minor bitching aside, there are three things I want to get to in this play, two of which will fit. These are the difference between jobs, procedures, & basic, as well as the screen format, but first a word about modems.

I have verified that a 1200 baud modem can be hooked up to the QL without a Modaptor using the following connections.

QL[SER2ir]		Modem
[DB-9]		[DB-25]
Pin		
1 - GND	------>	7 - Signal Ground
2 - TxD	------>	2 - TD
3 - RxD	------->	3 - RD
4 - No Connection		
5 - No Connection		
9 - +12v	------>	20 - DTR
9 - +12v	------>	4 - ???

I have used the above connections with an Avatex 1200 modem. Incidentally, the pins 6,7,8 on the North American QL DB-9 are tied to ground. [That info is verified by Wilf Rigter.] This is not mentioned in the QL User Guide presumably because in the UK a British Telecom type of connector with 6 pins is used.

****

Jobs. This can be confusing, particularly if you are not disposed to the hacking mentality in the least. However, I know some people have got it bass ackward, so I will attempt to unravel...and throw in a curve or two, as clarity demands.

Superbasic is a job. Superbasic is a special kind of a job, the root job and as such it always exists. It might be suspended, ie. not running or taking any processor time, but it is always there.

A superbasic program which you type in is not a job. The program itself is data which the superbasic job interprets.

However, if you take your superbasic program and compile it with Supercharge, Q_Liberator or Turbo the output of that process is a job.

The Psion quartet are all jobs. In general jobs are started with EXEC, EXEC_W, EX or EW. Each job has a distinct ID number allocated to it by QDOS. This number consists of a tag which is the nth job which QDOS has created and an offset in the Job table which QDOS maintains. You can switch between QL jobs by typing <CTRL><C>. You can change this character by poking the System Variable SV.CQCH, which stands for Change Keyboard Queue Character. This is a Word value at $28092 or Decimal 163986.

A procedure in Superbasic is the code you put between:

```
DEFine PROCedure Such_and_Such
:
END DEFine Such_and_Such
```

You can also define a Machine Language Procedure with the following format:

```
Word : Number of Procedures
Word : Pointer to Routine
Byte : Length of Name
Bytes: Characters of the name
Word : 0    [ a Seperator ]
Word : Number of Functions
Word : Pointer to Routine
Byte : Length of Name
Bytes: Characters of the Name
Word : 0    [ End of list ]
```

There is an example of this format in actual code below. Note that Machine Language Functions can also be defined. This is what makes Superbasic extensible, a unique & wonderful feature.

Here is the first tricky bit. We have seen that machine language can be a Procedure, a Job, but did you know it could also be something else...a Task? A task as defined by QDOS is an piece of asynchronous code executed to take care of some operating system business. An example is checking the status of a Floppy Disk Controller chip & taking appropriate action. The code can be started on interrupt or on the scheduler loop. It would seem then that conceptually all of the operating system stands apart from the Jobs/Procedure structure.

Another tricky bit here is the fact that these terms are used differently by most operating system writers. Unix talks of Processes, the Amiga talks of Tasks, the QL talks of Jobs and they all have different curves to them.

Machine language procedures can do anything. In particular, if I can raise the topic without being confusing, they can start jobs. This sort of situation lets you do something like just type 'EDITOR' and start an editor job. Here is an example of such code:

```
*
**************************************************************************************
*       This code Initializes the PROCEDURE "EDITOR"
*
INIT_PROC       LEA     PROCLIST,A1                   * POINTER TO PROC DEF LIST
                MOVE.W   BP_INIT,A2
                JSR     (A2)
                MOVEQ   #0,D0                         * NO ERRORS
INIT_EXIT       RTS
*
PROCLIST        DC.W    1                             * NUMBER OF PROC
                DC.W    EDIT_PROC-*                   * POINTER TO PROC
                DC.B    6                             * LEN.B
                DC.B    'EDITOR'
                DC.W    0                             * END OF PROC
                DC.W    0                             * NUMBER OF FUNCTIONS
                DC.W    0                             * END OF LIST
*
**************************************************************************************
*       This is the PROCEDURE "EDITOR"
*
EDIT_PROC       MOVEQ   #0,D1                         * BASIC IS JOB OWNER
                MOVE.L  #2048,D2                      * CODE LEN.L
                MOVE.L  #0,D3                         * DATA LEN.L
                SUBA.L  A1,A1                         * START ADDRESS OR 0
                MOVEQ   #MT_CJOB,D0                   * QDOS CREATE JOB
                TRAP    #1
                TST.L   D0                            * RETURNS A0=BASE,D1=JOB ID
                BNE.S   ABORT_PROC                    * IF ANY ERROR: ABORT

                MOVE.W  #$4EF9,(A0)+                  * JMP.L                2
                LEA     EDITOR,A1
                MOVE.L  A1,(A0)+                      * TO EDITOR            4
                MOVE.W  #$4AFB,(A0)+                  * QDOS JOB FLAG        2
                MOVE.W  #$000C,(A0)+                  * LEN.W                2
                MOVE.L  #'EDIT',(A0)+                                        4
                MOVE.L  #'OR 0',(A0)+                 * INSERT NAME          4
                MOVE.L  #'.99 ',(A0)+                                        4
*                                                                          ----
*                                                                        22
                MOVE.L  D1,(A0)+                * INSERT JOB ID           4
*                                                                        ----
*                                                                        26 BYTES
                MOVE.W  #DATA_LEN-1,D0          * DEFINED ELSEWHERE
DATAINIT        MOVE.B  #0,(A0)+
                DBRA    D0,DATAINIT
*
                MOVEQ   #0,D2                   * SET PRIORITY
                MOVEQ   #0,D3                   * TIMEOUT 0=EXEC : -1=EXEC_W
                MOVEQ   #MT_ACTIV,D0            * QDOS : MANAGER TRAP
                TRAP    #1                      * ACTIVATE  JOB
                BRA.S   EXIT_PROC
*
ABORT_PROC:     MOVE.W   UT_ERR0,A2             * REPORT D0 ERROR CODE TO CHANNEL 0 CONSOLE
                JSR      (A2)
EXIT_PROC:      MOVEQ   #0,D0
                RTS
*
*********************************************************************************
*       The Start of the Editor JOB code
EDITOR          ...

*
*********************************************************************************
*       How it works.

          This code would be loaded into memory with LBYTES filename,respr(16384)
        Then called from superbasic [ with CALL RESPR(0)]. That will cause the
        Procedure Initialize code to be executed, which will add the Machine Language
        Procedure EDITOR to the Superbasic Name List.
          Then when you type "EDITOR", the procedure creates the job with MT_CJOB.
        Note that the job doesn't yet start to execute. All QDOS does is reserve the
        memory for you and set up its various tables. The Trap returns the base of
        the allocated memory. In this memory you put the code
        JMP.L   Editor and the QDOS Job flag and standard name format.
          At this point, about all you have to do is start the job with MT_ACTIV.
```

Now for the screen format. The QL display has 2 modes, a 4 colour mode and an 8 colour mode. These are controlled from basic by the keyword MODE n, where n is 4 or 8.

The video ram in the QL normally consists of 32K bytes starting at $20000 or decimal 131072. The QL can have 2 screens of video ram, but the implementation of the dual screens is incomplete. The problem is that QDOS is locked into having the System variables start at $28000 & there is just no two ways around that, because it is hard coded in the ROM.

The QL screens are what you might call interleaved bit planes. In mode 4, the information is setup as follows:

```
Word 0               1               2               3
Byte 0       1       2       3       4       5       6       7
Bits 7654321076543210765432107654321076543210765432107654321076543210
     GGGGGGGGRRRRRRRRGGGGGGGGRRRRRRRRGGGGGGGGRRRRRRRRGGGGGGGGRRRRRRRR
```

The G stands for Green, the R stands for Red. You can play with the screen directly by poking a value into the video ram. A command such as POKE 131072,255 will set every green bit in the first word of video ram and draw a short green line in the upper left hand corner of the screen.

POKE 131073,255 will similarly set all the red bits.

POKE 131072,255 : POKE 131073,255 will set all the green and red bits and give you a white line. The fourth colour black is the absence of light with all bits clear ie. zero.

The very first pixel of the first line is controlled by bit 7 of byte 0 [for Green] and bit 7 of byte 1 [for Red]. The second pixel is similarly controlled by bit 6 of the two bytes and so on.

A screen resolution of 512 pixels thus requires 512/8 = 64 words [128 bytes].

In mode 8, the information is setup as follows:

```
Word 0               1               2               3
Byte 0       1       2       3       4       5       6       7
Bits 7654321076543210765432107654321076543210765432107654321076543210
     GFGFGFGFRBRBRBRBGFGFGFGFRBRBRBRBGFGFGFGFRBRBRBRBGFGFGFGFRBRBRBRB
```

F stands for Flash and B stands for Blue. Clearly with 3 bits controlling the colour there are 2^3 or 8 colours as you know. One wonders if the Sinclair engineers considered using 16 colours rather than implementing a FLASH.

A screen resolution of 256 pixels thus requires 256/4 = 64 words.

As usual, let me say that if you are a QL enthusiast feel free to write c/o the Editor.

This assembly language is a subroutine I developed to write directly to the screen in MODE 8.

>eof

```
*******************************************************************************
*
*  PLOT POINT DIRECTLY
*       SCREEN  128 BYTES WIDE ! 2 BITS BYTE0 & 2 BITS BYTE1 !
*               4 PIXELS / WORD ==> 256 PIXELS
*               256 ROWS DEEP   !
*
*   EXPECTS  D7 = COUNT
*            D0 = ROW (0-255)
*            D1 = COL (0-255)
*   RETURNS  D0 = ERROR CODE
*
*  WANT TO PUT 0-7 IN BITS RGB, WITH F CLEAR ALWAYS
*
*                    BIT7 BIT6 BIT5 BIT4 BIT3 BIT2 BIT1 BIT0
*     EVEN BYTE       G    F    G    F    G    F    G    F
*      ODD BYTE       R    B    R    B    R    B    R    B
*  OR
*  CARRY BIT X/C    15 14 13 12 11 10 9  8     7  6  5  4  3  2  1  0
*                   G  F  G  F  G  F  G  F     R  B  R  B  R  B  R  B
*
                    -  -                       -  -
```

```
*
PLOT_PNT        MOVEM.L D1-D3/D7/A0,-(A7)       * SAVE SOME REGISTERS
*
RANGECHK        TST.W   D0
                BLO     OR_ERROR
                CMPI.W  #256,D0                 * TEST IF ROW 0-255?
                BHS     OR_ERROR
                TST.W   D1
                BLO     OR_ERROR
                CMPI.W  #256,D1                 * TEST IF COL 0-255?
                BHS     OR_ERROR
*
                MOVE.L  #$00027F80,A0           * BOTTOM LEFT HAND CORNER OF SCREEN
*
                MULU    #128,D0                 * CALCULATE ROW DISTANCE IN BYTES
                SUBA.L  D0,A0                   * CALC LEFT HAND CORNER OF ROW
*
                MOVEQ   #0,D2                   * CLEAR TOP
                MOVE.W  D1,D2                   * COPY COLUMN
                LSR.W   #1,D2                   * /2 CALC BYTE # : 255-127 & 3 BELOW
                BCLR    #0,D2                   * MAKE SURE WORD BOUNDARY
                ADDA.L  D2,A0                   * A0=> APPROPRIATE WORD
                ANDI.W  #$03,D1                 * MASK OFF 0-3
* 4 CASES D1=0,1,2,3
                MOVEQ   #0,D3                   * TARGET MASK
                ANDI.W  #$07,D7                 * MASK OFF COUNT 0-7
*
                TST.W   D1
                BNE.S   TEST1
EQUALS0         MOVE.W  #$3F3F,D2               * CLR MASK
                LSR.B   #1,D7                   * BIT0 TO X/C
                ROXR.W  #1,D3                   * X/C TO BIT15
                LSR.B   #1,D7                   * BIT1 TO X/C
                ROXR.B  #1,D3                   * X/C TO BIT7
                LSR.B   #1,D7                   * BIT2 TO X/C
                ROXR.B  #1,D3                   * X/C TO BIT 7, BIT7 TO BIT6
                BRA.S   PUT_DATA

TEST1           CMPI.W  #$01,D1
                BNE.S   TEST2
                MOVE.W  #$CFCF,D2               * CLR MASK
                LSR.B   #1,D7                   * BIT0 TO X/C
                ROXR.W  #3,D3                   * X/C TO BIT13
                LSR.B   #1,D7                   * BIT1 TO X/C
                ROXR.B  #1,D3                   * X/C TO BIT7
                LSR.B   #1,D7                   * BIT2 TO X/C
                ROXR.B  #3,D3                   * X/C TO BIT5, BIT7 TO BIT4
                BRA.S   PUT_DATA
TEST2           CMPI.W  #$02,D1
                BNE.S   TEST3
                MOVE.W  #$F3F3,D2               * CLR MASK
                LSR.B   #1,D7                   * BIT0 TO X/C
                ROXR.W  #5,D3                   * X/C TO BIT11
                LSR.B   #1,D7                   * BIT1 TO X/C
                ROXR.B  #1,D3                   * X/C TO BIT7
                LSR.B   #1,D7                   * BIT2 TO X/C
                ROXR.B  #5,D3                   * BIT7 TO BIT2,X/C TO BIT3
                BRA.S   PUT_DATA
TEST3           CMPI.W  #$03,D1
                BNE     NC_ERROR
                MOVE.W  #$FCFC,D2               * CLR MASK
                LSR.B   #1,D7                   * BIT0 TO X/C
                ROXR.W  #7,D3                   * X/C TO BIT9
                LSR.B   #1,D7                   * BIT1 TO X/C
                ROXR.B  #1,D3                   * X/C TO BIT7
                LSR.B   #1,D7                   * BIT2 TO X/C
                ROXR.B  #7,D3                   * BIT7 TO BIT0,X/C TO BIT1
PUT_DATA        AND.W   D2,(A0)                 * G/F/R/B TO 0'S
                OR.W    D3,(A0)                 * INSERT DATA
OK_EXIT         MOVEQ   #0,D0
PP_EXIT         MOVEM.L (A7)+,D1-D3/D7/A0       * RESTORE REGISTERS
                RTS                             * RETURN WITH ERROR CODE
NC_ERROR        MOVEQ   #-1,D0                  * NC_ERR = -1
                BRA.S   PP_EXIT
OR_ERROR        MOVEQ   #-4,D0                  * OR_ERR = -4
                BRA.S   PP_EXIT
*
*
```

****

```
   1 REM program reprinted from
March'87 issue of the newsletter
of SLUG......Sinclair Louisville
Users Group
   2 REM ...don't panic when you
 run the listing...just wait.
   3 REM a colour TV or monitor
gives even more impressive resul
ts.
  10 LET a=USR "p"
  20 FOR j=0 TO 31: READ b: POKE
 a+j,b: NEXT j
  30 DATA 6,255,197,6,75,62,6,21
1,254,16,252,193,16,236,201
  40 DATA 33,0,10,43,126,211,254
,6,205,5,32,-3,175,132,200,24,-1
4
  50 RANDOMIZE USR 65511
```

****

--- Computer Industry Advertising Definitions:

NEW-different color from previous model.
ALL NEW-no interchangeable parts with previous model.
IMPROVED-old bugs replaced with new ones.
EXCLUSIVE-imported product.
UNMATCHED-almost as good as the competition.
FOOLPROOF-no provisions for adjustment.
ADVANCED DESIGN-ad copy writer doesn't understand how it works.
FIELD TESTED-manufacturer lacks test equipment.
FACTORY DIRECT-manufacturer in fight with distributors.
RUGGED-too heavy to move.
LIGHTWEIGHT-lighter than rugged.
PORTABLE-has a handle.
HIGH PERFORMANCE-almost meets designer specs.
BREAKTHROUGH-we finally figured out how to sell it.
EFFICIENT-uses 1% less power than previous model.
IBM COMPATIBLE-paint matches IBM PC.
MAINTENANCE FREE-impossible to repair.
SATISFACTION GUARANTEED-ours, on receipt of your check.
RELIABLE-prototype worked at least 1 week between repairs.
FULL SUPPORT-broken units available as spare parts.
OBSOLETE-dependable, reliable, inexpensive and readily available.

Reprinted from the May/87 issue of the N/L of the Dallas T/S/A Users Group

A New Life for Sinclair/Timex Computers

How to convert a Sinclair ZX81 or Timex-Sinclair TS1000 into a programmable printer buffer

By R.L.L. Hu & J.J. Chang

Not too many years ago, more than a million very-low-cost Sinclair ZX81 and Timex-Sinclair TS1000 computers were sold to consumers who wanted a taste of working with the new devices. Most of these computers are now stored away, just gathering dust due to obsolescence. If you're lucky enough to have one of these tiny mites but haven't used it in years and don't know what to do with it, here's a way to convert it into a Centronics-compatible parallel printer buffer that can free up your present computer during printing operations.

As an example of how effective the modification is, this manuscript tied up my computer for 11 minutes while printing. In contrast, using the ZX81/TS1000 as a printer buffer reduced this to 12 *seconds*!

You can do the conversion at a fraction of what it would cost you to buy a commercial buffer or build one from scratch. Converting a ZX81 or TS1000, as described here, offers a number of advantages not normally obtained with commercial or home-built buffers. Among these are: a power-on memory test/size-determination feature; a Multi-Copy Mode that lets you select up to 255 copies of a document to be printed; continuous display of buffer filled or copies remaining to be printed; a pause function for suspending printing; a clear key to reset the Buffer; and automatic cancelation of Multi-Copy Mode on buffer size overrun.

This project makes *permanent* hardware modifications to the computer to make it possible for all the printer buffer interfacing logic to be contained in the computer's original case. Once the modifications are made, you will no longer be able to run any of the computer's original software, of course.

About the Circuit

To transform the ZX81 and TS1000 from computer into Printer Buffer, the original r-f modulator and ULA (uncommitted logic array) and ROM chips must be removed. A 2716 EPROM programmed with the Printer Buffer Program (see listing) goes into the original ROM socket and then a small interface board with ICs and cables that go to the printer and your present computer plugs into the original ULA chip socket. A 1-digit hexidecimal display, also on the interface board, shows the status of the Printer Buffer. Once the modification has been performed, functions such as pause, multiple copies and clear-buffer are provided through the ZX81's or TS1000's keyboard.

Shown in Fig. 1 is the schematic diagram of the interface-board's circuitry that converts the ZX81 and TS1000 into the Printer Buffer. The two halves of the 74LS139 dual 1-of-4 decoder used for *IC1* decodes both the memory and I/O (input/output) space. Quad D flip-flop *IC2* latches hexidecimal decoder/driver/display *IC5*. The TIL311 used for *IC5* contains all decoding and driving circuitry as well as the LED hex display itself in a single IC package. The left and right decimal points (see Fig. 2), for example, are used as extensions of the contents of 4-bit hex display *IC5*, representing the fifth and sixth bits, respectively. As a result, the possible range of the display is from 0 to 63, displayed as "0" through ".F.".

Programmable peripheral interface *IC3* handles all electrical interfacing required between computer and printer and monitors the Buffer's keyboard for operator commands. The system clock is derived from the TLC555 timer chip, shown as *IC4* in Fig. 1. The TLC555 timer was chosen for this application because it has a free-running clock frequency of up to 2 MHz. The standard 555 timer's clock frequency is much more restrictive and, thus, is not adequate for this application.

Of the 64K bytes of possible memory space available in the ZX81 and TS1000, the first 16K is allocated for ROM, while the second 16K is allocated for the standard 16K RAM pack. Though the third and fourth 16K spaces are not used by Sinclair, other manufacturers have built RAM packs that make use of these spaces. The Buffer's hardware and software have been set up to accommodate this maximum of 48K of user RAM, but be advised that the project has *not* been tested with such RAM packs.

Construction

Two main procedures must be performed to convert the ZX81 and TS1000 from a computer into a Printer Buffer. First you must modify the computer's motherboard to accommodate the preprogrammed 2716 EPROM and the interface board. Then you build the interface board and install it in the computer.

When you disassemble the ZX81 or TS1000, make certain to remove the three screws hidden under the rubber feet. If you work carefully, there should be no need to remove the two keyboard ribbon cables from their connectors. If you do remove them, however, be *very* careful when putting them back.

Once you have the computer open, remove the 8K ROM and ULA chips from their sockets. Then remove the r-f modulator and tie all anode leads of *D1* through *D8* to ground.

Now is an excellent time to program the EPROM you'll be using in this project. The hexidecimal code for the Printer Buffer program is given in the Program Listing. When the Printer Buffer is operating, the amount of information accumulated at any instant is shown on the hex display as a whole number of kilobytes. (Actually, 1 KB = 1,024 bytes). Therefore, 1,500 bytes will be displayed as a "1". If you prefer to round the number up to the next kilobyte so that it is displayed as a "2" (kilobytes implied), make the changes indicated in the Program Listing when you program the data into the EPROM. (If you don't have an EPROM programmer, an excellent stand-alone model that handles up to 128K EPROMs can be built from plans featured in the February and March 1987 issues of *Modern Electronics*—Editor)

To make the 2716 EPROM work in the original ROM socket, pin 18 (-CE) must be tied to ground and pin 21 (V_{pp}) must be tied to +5 volts. Alternatively, you can bend pins 18 and 21 on the 2716 itself away from the IC case, install the EPROM in the ROM socket, and solder the pins to ground and +5 volts via lengths of hookup wire.

If you decide to cut traces, keep in mind that you may also be cutting off signals that are needed elsewhere on the motherboard. In such a case, jumper wires will have to be used to reconnect the isolated sections back into the circuit.

Cut a piece of perforated board with holes on 0.1" centers to the size and shape shown in Fig. 3. The board and component layout shown will fit both the ZX81 and TS1000 cases and motherboards. Any construction technique that will yield a low-profile wired modular assembly will do. The prototype shown in the photos was built using 3M's Scotchflex Breadboarding System. Other choices include Vector Electronics' wiring pencil, printed-circuit board layout, etc. In all cases, use low-profile sockets for all ICs.

Using sockets for all ICs and for plugging in the DIP connectors at the ends of the Centronics cables, wire the circuit exactly according to Fig. 1. All signals from top to bottom on the left side of the schematic diagram are from the original ULA chip socket, at the pin numbers indicated. You need adapter pins that can be soldered to the interface board and be plugged into the ULA chip socket in the computer. Square header pins are too large for this purpose and are *not* recommended. If you can't find appropriate pins, try using pins removed from ribbon cable DIP connectors.

Plug the wired circuit board module into the ULA socket on the ZX81's or TS1000's motherboard. Make sure you plug it in so that the added pins line up exactly with the slots in the ULA socket and pins 1 through pin 40 are properly indexed. Then connect the cables from the interface board to your printer and the computer that will serve as the host. If your computer printer port doesn't use the eighth data bit, be sure to ground the appropriate pin going into the Printer Buffer. Also, signal lines such as Paper Out, Error, etc., should be tied to the appropriate logic level if your printer controller interrogates these lines.

Without any RAM packs installed in the Buffer, power up the Buffer, printer and computer. Internal memory (1K in the ZX81 and 2K in the TS1000) is sufficient for testing the Printer Buffer. Go through all functions (see Operation below). Once the Buffer has tested okay, power down the system, plug in the RAM pack, power up the system again and repeat all function tests.

Before reassembling the modified ZX81 or TS1000, cut two holes in the case. Cut one hole in the top of the case to provide a window to view the hex display. Cut the other hole in the side for the printer cables. Make provisions for and install a strain relief for the printer cables as well.

Using the Buffer

Keyboard control for full operation of the Printer Buffer is provided by the "C" and "S" keys on the ZX81 and TS1000. These keys control the Clear and Start/Stop/Set Copy functions, respectively. Pressing the "C" key at any time resets the Buffer, which cancels in its entirety all settings and any print job in progress.

On power-up, the Buffer runs through a simple memory test and determines the amount of memory available. The display indicates the current kilobyte of memory being tested. With the 16K RAM pack, the display will cycle up from 0 to F and then pause briefly prior to becoming ready for use. The Buffer is now in the Buffer Mode.

- *Buffer Mode.* In this mode, the Buffer simply accepts data from the host computer and sends it to the printer. This is the default mode. The Buffer will always be in this mode unless the Multi-Copy Mode is selected. Hence, immediately after power-up, pressing "C" to clear the Buffer, or termination of a multi–copy job, this is the active mode.

Since the output speed of the host computer is considerably faster than the printing speed of the printer, data will accumulate in the Printer Buffer. The amount of accumulation at any moment in time is shown on the hex display in kilobytes.

In the Buffer Mode, pressing the "S" key temporarily suspends the printing operation. During suspension, the display flashes on and off, but data from the host computer will be accepted if the Buffer is not full. Pressing the "S" key once more resumes the printing operation.

- *Multi-Copy Mode.* This mode allows a block of data to be printed up to a maximum of 255 times without intervention from the host computer. This mode can be selected only after a power-on or pressing of the "C" key to clear the Buffer and while the Buffer is waiting for the first character from the host computer. The number of copies to be printed is then set.

Should any characters be received from the host before the number of copies is set, the multi-copy option expires and the Buffer reverts to its default Buffer Mode. If this occurs, you must turn the Buffer off and then on or press the "C" key to clear the buffer and then reenter the Multi-Copy mode.

You select the number of copies to be printed by pressing the "S" key once for each additional copy desired. Each time the "S" key is pressed, the display increments to indicate the number of copies set. The largest number that can be displayed unambiguously by the display is 63, which is shown as ".F.". Consequently, if the number of copies to be printed exceeds 63, the display restarts from 0. However, the number of copies is not returned to 0, and the internal counter continues to register the correct number of copies, up to 255 maximum.

After selecting the number of copies, you instruct the host computer to download whatever is to be printed to the Buffer. Once data transfer has begun, the display will show the number of kilobytes transferred. When downloading is complete, press the "S" key to commence multi-copy printing. At this time, the display will go back to showing the number of copies to be printed, including the current copy being printed.

Should the Pause/Copy key be pressed prior to completion of downloading, multi-copy printing will commence immediately based only on that information already stored in the Buffer.

If during downloading the information exceeds the capacity of the Printer Buffer, the Buffer automatically switches back to Buffer Mode and printing commences immediately. Only one copy will be printed.

Once printing has commenced, the Printer Buffer will not accept any data from the host computer until the preset number of copies have been run out.

Conclusion

This reworking of the basic ZX81/TS1000 gives new life to these computers. With just a few dollars and a few hours of your time, you'll restore an idle computer to useful service and more than recoup whatever monetary investment you made for it in the first place.

PARTS LIST

Semiconductors:

IC1 - 74LS139 dual 1-of-4 decoder
IC2 - 74LS175 Quad D flip-flop
IC3 - 8255A-5 programmable peripheral interface
IC4 - TLC555 CMOS timer
IC5 - TIL311 hexidecimal decoder/driver/display

Capacitors:

C1 - 200 pF disc
C2 - 10 F, 16-volt tantalum

Resistors (1/4 - watt, 5%):

R1,R2 - 820 ohms
R3 - 100 ohms
R4 - 330 ohms

Miscellaneous:

Programmed MM2716-45 2K x 8 450-ns EPROM (see text); perforated board with 0.1" hole centers(Radio Shack *276-158 or similar); low-profile sockets for all ICs and plug-ins for Centronics cables; adapter pins for plugging into sockets (see text); male and female Centronics-compatible printer cables with 14-pin DIP connectors at other end; hook-up wire; solder; etc.

NOTE: The source-code listing for the Printer Buffer Program is available in printed form from NAND Engineering, 1458 Meadowbrook Rd., Gloucester, Ontario, K1B 5G7 for 18.00 prepaid.

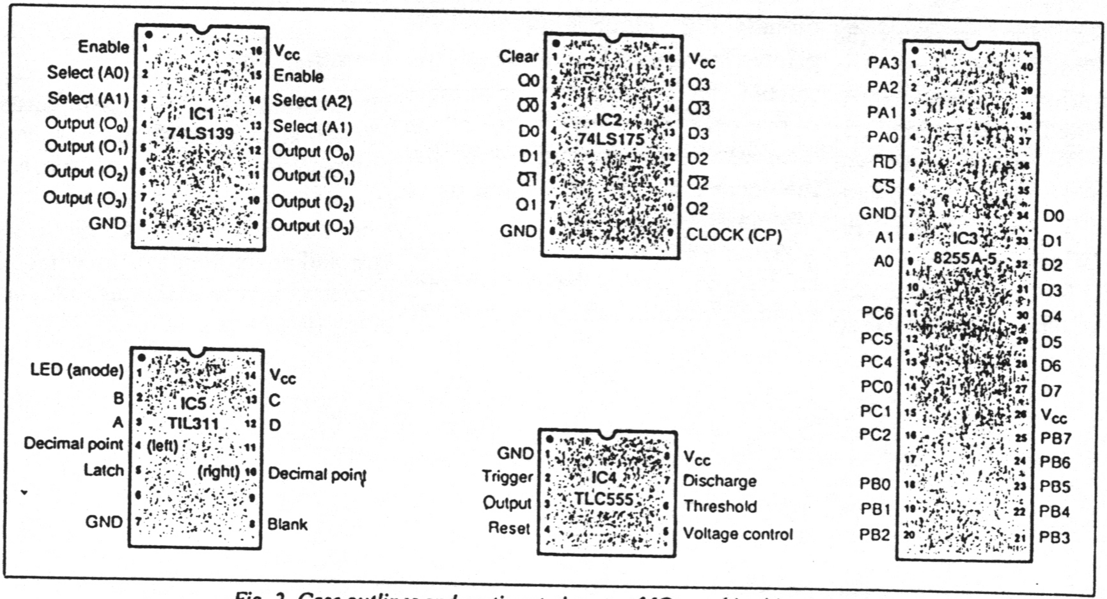
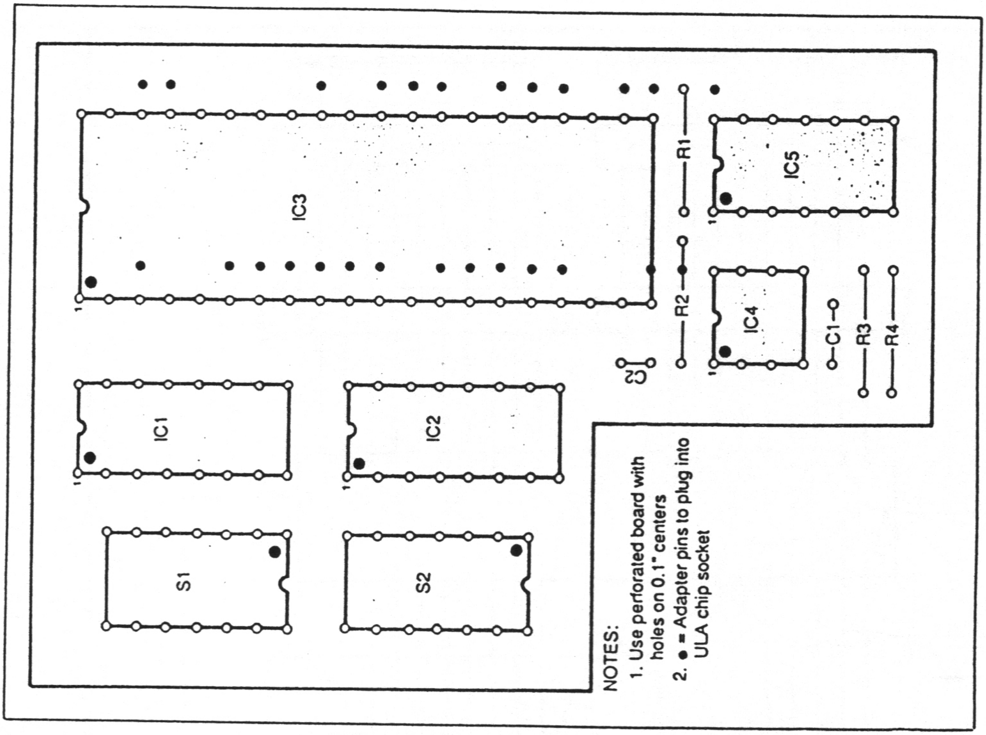
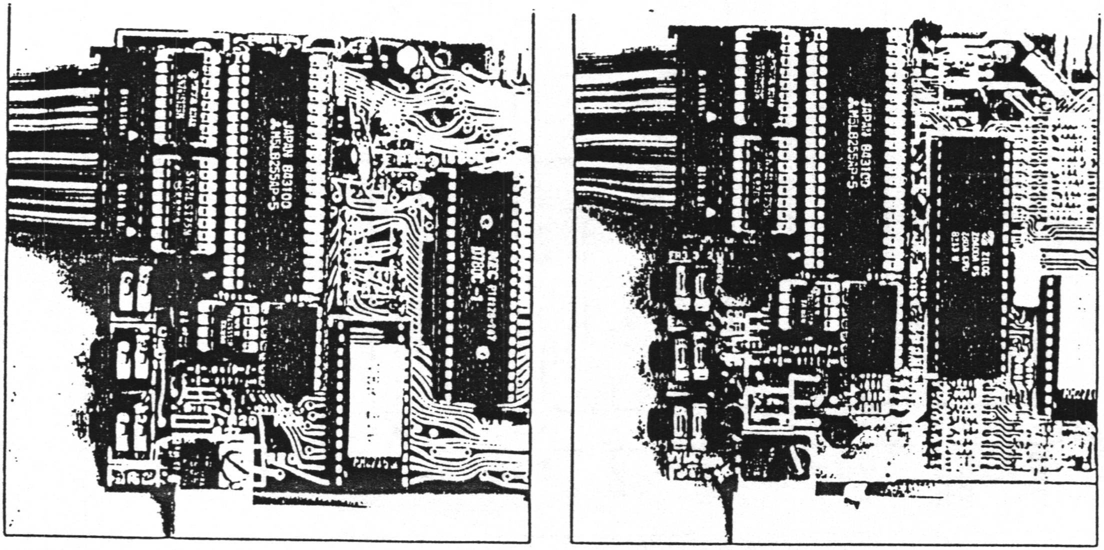


Fig. 2. Case outlines and pertinent pinouts of ICs used in this project.



NOTES:
1. Use perforated board with holes on 0.1" centers
2. ● = Adapter pins to plug into ULA chip socket

Fig. 3. Plug-in circuit-board module size, shape and component layout guide. Use board with holes on 0.1" centers and low-profile sockets with all ICs and for Centronics cable plug-ins. Refer to Fig. 1 for wiring details.



Modification board installed in Sinclair ZX81 (left) and Timex/Sinclair TS1000 (right) computers. Keyboard is at right, and left side of computer is at bottom in both photos. Computer and printer cables plug into board as shown and should be secured with a strain relief. Note that boards do not interfere with other components inside computers.

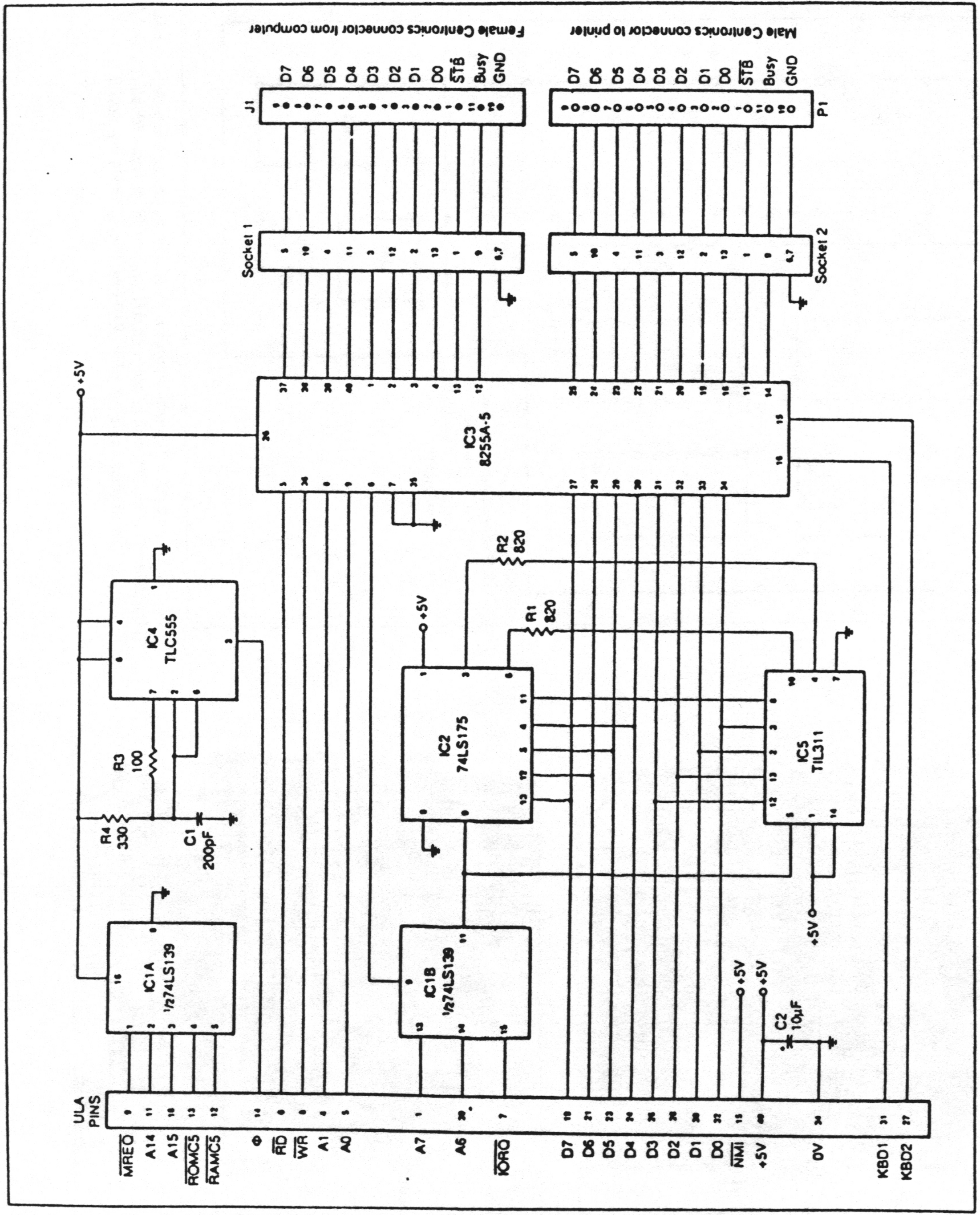


Fig. 1. Complete schematic diagram of modification module

originally appeared in the Aug/85
C.A.T.S. N/L and is reprinted from the Sept/85
TIMELINZ N/L.

TS 1000 Notes

1. To initialize without pulling the plug, enter RAND USR 0.

2. Program size PEEK 16396 +256 * PEEK 16397 - 16509

3. Variables size PEEK 16404 + 256 * PEEK 16405 - PEEK 16400 - 256 * PEEK 16401

4. Display size PEEK 16400 + 256 * PEEK 16401 - PEEK 16396 - 256 * PEEK 16397

5. VAL "4E4" uses 8 bytes, 4E4 uses 9 bytes; VAL "40000" uses 10 bytes, 40000 uses 11 bytes.

6. BOTH PRINT and LPRINT delete the leading space of a token when used at the beginning of a PRINT, AT, or TAB statement as well as when followed by another token.

7. To display the bottom two lines use within a program: (line number) POKE 16418,0. Before using SCROLL or INPUT switch off with: (line number) POKE 16418, 2.

8. To print the bottom two lines enter line 1 REM shift J, shift V, function LN, shift C, graphic shift A, function TAN, then POKE 16517,107. Within the program use (line num) RAND USR 16514, then (line num) COPY.

9. With MC programs stored in 1 REM, POKE 16510,0 to change the REM number to 0, and protect it from accidental deletion.

10. To make a line of MC invisible (certain codes will self-destruct when displayed) POKE 16514, 118. Remember the MC now begins at 16516.

11. To make a program run automatically when loaded, SAVE the program with a line statement (usually at the beginning or end of a program). Example: 9010 SAVE " MET" ; 9020 RUN. Now start the tape recorder and enter GOTO 9010. It will run automatically the next time it's loaded.

12. To stop a program from automatically running after loading, go to FAST mode then enter PRINT USR 836. The machine will go into a loading state. Start the tape recorder. When the program has completed loading an error code will appear. Hit enter and the program will LIST.

13. To adjust a tape recorder's volume and/or verify if a program is loading properly, simply stop the tape recorder a few seconds after you have begun to load a tape. The computer will either crash, which means the program was loading correctly, or will continue in a search pattern which means there was a problem with either the title or low volume.

##############################

MonroeRLE

A Review of TECH DRAW JR. from Zebra Systems, Inc.
By Syd Wyncoop

Tech Draw is a joystick controlled version of Zebra's Koala pad drawing software. I find it much better, in that you do not get the stray pixels on the screen that the Koala pad generates. There is also a huge difference in price, since you are not required to purchase the Koala pad.

The first thing you notice about the program, as it is packaged, is the instruction manual. It is the same one that was sold with the Koala pad; therefore must be interpreted in light of a joystick's operation. Also, the manual is barely legible.

Tech Draw will enable you to design some very fine black and white screen displays; however, it desperately needs color enhancements. Why design a graphics package for a color computer without color capability?

There is a choice of 16 brushes to draw and are chosen from pull down menus. These menus are accessed from the main menu which is always on the lower 2 screen lines. The brush control is from one pixel to an entire character block.

There are 35 built-in shades which can be drawn with, but are better utilized as fill patterns. Any fully enclosed shape can be filled in with any of these patterns to generate some very nice pictures.

There is the usual undo command, in case you don't like the results of the last command, as well as a clear-the-screen option if you would like to start over fresh.

The cursor is difficult to position with any accuracy due to the speed it moves. To correct this requires use of the arrow cursor keys. It should have been possible to initially move the cursor slowly, speeding it up after a few continuous pixels are crossed. There are tracking options, to correct for sideways movement of the joystick. They will only read the vertical or horizontal movement, as selected, no matter how far you stray. This is nice for drawing straight lines and of course can be turned off to allow for diagonal lines.

There are three character fonts and three sizes of each for lettering your drawings. You can print to any spot on the screen, instead of only on the normal print lines. The routines to move the cursor, when in text mode, are intolerably slow. The manual states this is to allow exact positioning; however, you can nap while waiting for the cursor to cross large areas of the screen. These routines shold also start slow and speed up as the position of the joystick is held.

Lastly, the I/O menu does not interface with any storage media other than tape. While it is unreasonable to expect Zebra to include routines for all the various systems on the market, they could have left this interface in BASIC, where it could be easily modified to suit your needs.

I think the program needs some major improvements to be of much value. Tech Draw is definitely not a finished product, although it is being marketed as such.

////////////////////////////////###\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

Reprinted from the Oct-Nov/86 issue of the P.A.T.S. User Group newsletter

TIMEX TIPS

By Chuck Dawson

QUESTION: How do you go about specifying certain keys that we do not have on the Sinclairs? There is no ESCAPE or CONTROL keys on the keyboard, but it is used in Mterm. How can we use it in BASIC programs?

ANSWER: People tend to be intimidated by keyboards, especially if there seem to be many more keys than on a standard typewriter. The truth is, since we are limited to eight binary digits, there are only 256 possible combinations of ones and zeros. The T/S keyboard has only 40 keys, and yet it can send every one of these 256 combinations. Some, of course, are accessed by first getting the 'F' or 'E' or 'G' cursors. If you look in the manual at the table of characters, you will see that only a few codes near the beginning are 'not used'. These are usually used to give commands to a printer. For example, most printers use a '12' to advance the paper to the top of the next sheet. Many use '15' to set the condensed type. Most use the ESCAPE code (27) plus another number to do a variety of other things. In BASIC, this would take the form LPRINT CHR$ 27 or LPRINT CHR$ 15. The Mterm terminal emulation program produces an ESCAPE by holding down SYMBOL SHIFT and pressing ENTER. Another computer may have an ESCAPE key to produce the same results. By the ASCII convention, printable characters do not begin until 32, so those early codes that are used to control printers are called 'Control Codes'. Some computers have a CONTROL key which gives access to those codes by holding down the CONTROL key and pressing another key. A '1' is Control A, a '2' is Control B, and so forth. In BASIC, these would be represented by CHR$ 1, CHR$ 2, and so forth. The T/S uses many of the codes above 127 for its KEYWORDS. For example CHR$ 180 is the TAN function on the 2068 and an inverse '0' on the 1000. That is why, when machine code is poked into a REM statement, you sometimes see letters, keywords, graphic characters, and so on. The computer is faithfully displaying each code as it has been programed to do. So there you have it. Just think of each keystroke as sending a number from 0 to 255 to the computer (or to the Modem or to the Printer) and suddenly things don't seem quite as complicated as they at first appeared.

Reprinted from the DATA EXPANSION, the N/L of the Ft.Worth T/S Users Group

```
   1 REM PERPETUAL CALENDAR 2068
         ©1984 BY I. AUERSBACHER

   2 POKE 23609.20· POKE 23658,8
   8 BORDER 5: BRIGHT 1: CL3 : D
IM M$(12.9)
   9 BEEP .1,30: GO SUB 1000: CL
S
  10 PRINT ' PAPER 6;" *** CALEN
DAR (1533-9999 AD)*** "
  12 RESTORE : FOR Z=1 TO 12: RE
AD M$(Z): NEXT Z
  15 LET Z$=" 1 2 3 4 5 6 7 8 91
0111213141516171819202122232425 2
62728293031": BEEP .05,20
  18 INPUT "Y-YEAR. M-PRINT MO,
  D-DISPLAY:"; LINE T$
  19 IF T$="Y" THEN GO TO 30
  20 INPUT "month (1-12):";A: IF
 A<1 CR A>12 THEN BEEP .5,-20: G
O TO 20
  30 INPUT "YEAR (YYYY):";B: IF
B<1582 OR B>9999 THEN BEEP .5,-2
0: GO TO 30
  35 CL3
  40 IF T$="Y" THEN FOR X=1 TO 1
2: LET A=X
  60 LET B=INT B: LET E=B-(A<3)
  65 LET C=A+12*(A<3)+1
  70 LET D=INT (E/400)-INT (E/10
0)+INT (1.25*E)+INT (2.6*C)
  75 LET D=D-(7*INT (D/7))+1
  80 IF (A=4)+(A=6)+(A=9)+(A=11)
 THEN LET C=30
  85 IF (A=1)+(A=3)+(A=5)+(A=7)+
(A=8)+(A=10)+(A=12) THEN LET C=3
1
  90 IF A<>2 THEN GO TO 200
 100 LET C=28: IF (B/4=INT (B/4)
)*(B/100<>INT (B/100))+(B/400=IN
T (B/400)) THEN LET C=29
 200 IF T$="Y" THEN CL3
 202 PRINT " ********************
***********"
 205 PLOT 10,132: DRAW 225,0: DR
AW 0,-112: DRAW -225,0: DRAW 0,1
12
 210 PLOT 10,118: DRAW 225,0
 220 PRINT AT 6,2:"SU  MO  TU  W
E  TH  FR  SA": PRINT AT 3.0;
 230 FOR Z=1 TO 6: PLOT 10,118-(
Z-1)*16: DRAW 225,0: NEXT Z
 240 FOR Z=1 TO 6: PLOT 40+(Z-1)
*32.132: DRAW 0.-112: NEXT Z
 245 IF T$="Y" AND A=1 THEN LPRI
NT  "[redacted]  ";B;" [redacted]
": LPRINT : LPRINT
 247 IF T$="Y" THEN PRINT TAB 12
;M$(A): GO TO 264
 252 PRINT TAB 8;M$(A);TAB 18:B
 264 DIM C$(74): LET J=8: LET K=
2: LET C$=C$( TO 2*D-2)+Z$
 270 FOR Z=1 TO C+D-1: PRINT AT
J.K:C$(2*Z-1 TO 2*Z): LET K=K+4
 280 IF (Z/7=INT (Z/7)) THEN LET
 J=J+2: LET K=2
 290 NEXT Z
 295 IF T$="Y" THEN COPY : NEXT
X
 296 IF T$="M" THEN COPY
 299 INPUT "ANOTHER YEAR OF MONT
H? (Y/N):";Y$: IF Y$<>"N" THEN R
UN
 300 DATA "JANUARY","FEBRUARY","
MARCH","APRIL","MAY","JUNE","JUL
Y","AUGUST","SEPTEMBER","OCTOBER
","NOVEMBER","DECEMBER"
 310 STOP
```

```
1000 CL3
1010 PRINT "TO PRINT THE FOLLOWI
NG PRESS:"
1015 PRINT : PRINT
1040 PRINT "L--- To Lprint"
1045 PRINT "C--- ""THIS CALENDAR
 IS FOR:"""
1050 PRINT "M--- ""MONTH of BIRT
H"""
1060 PRINT "Y--- ""YEAR of BIRTH
"""
1070 PRINT "N--- If you don't wa
nt to Lprint"
1120 IF INKEY$="M" THEN GO TO 40
00
1130 IF INKEY$="Y" THEN GO TO 50
00
1140 IF INKEY$="L" THEN GO TO 60
00
1150 IF INKEY$="N" THEN RETURN
1157 IF INKEY$="C" THEN GO TO 85
00
1160 GO TO 1100
4000 LPRINT : LPRINT "          M
ONTH of BIRTH": LPRINT : GO TO 1
100
5000 LPRINT : LPRINT "          Y
EAR of BIRTH": GO TO 1100
6000 CLS : PRINT : PRINT : PRINT
 "Do you want the Lprint CENTERE
D?"
6010 IF INKEY$="Y" THEN GO TO 61
00
6020 IF INKEY$="N" THEN GO TO 63
00
6030 GO TO 6010
6100 PRINT : PRINT "Enter up to
32 characters."
6110 INPUT Z$: IF LEN Z$>32 THEN
 GO TO 6110
6150 LET L=(32-LEN Z$)/2
6160 LPRINT TAB INT L;Z$
6165 LPRINT ''
6170 GO TO 1000
6300 PRINT : PRINT "Enter up to
32 characters."
6310 INPUT Z$: IF LEN Z$>32 THEN
 GO TO 6310
6350 LPRINT Z$
6355 LPRINT ''
6360 GO TO 1000
8500 LPRINT : LPRINT "    THIS
CALENDAR IS FOR:"
8510 LPRINT : GO TO 1000
9999 SAVE "calendar" LINE 2
```

This program originally appeared in CTM magazine and has been reprinted in numbers of other newsletters.

This article originally appeared in QUANTA, the N/L of the British QL Users Group by the same name, and is reprinted from the June/87 issue of the N/L of The St.Louis T/S Users Group.

QL REAL-TIME CLOCK

by Roy Barber

I wonder how many QL owners are aware of the fact that their QL may already contain a working real - time clock which, with the aid of a battery and a couple of diodes, may well give the battery - backed clock so many have requested.

The ZX8302 chip in the QL already contains all the circuitry required for maintaining the time and date when the QL is switched off, except for a battery and couple of diodes. Sinclair obviously intended to provide the battery back-up as the board has provision for at least one of the diodes, but this has been replaced by a wire link. It is easy to fit the necessary components and details of how to do this are provided below later.

I suspect Sinclair did not incorporate the battery back-up as it shows some lack of reliability on some machines and he did not want any more returned for service than he was already getting. Some of my friends and I fitted batteries and diodes to five QLs with reasonable success. Of the five, two appear to work 100% reliably, one about 98%, and the remaining two about 75% of the time. It appears that the failures which do occur happen on power up or down. I have monitored the crystal oscillation while the chip is under battery power and it seems absolutely reliable on all the QLs. With a large capacitor across the battery supply, only a slight change in mean level of the oscillator can be observed. From this I presume failure to maintain time is due to garbage getting into the clock registers due to a lack of isolation of the master chip during the power up or down phase. If a way of deselecting the chip during switch on or off could be derived it should work 100%. To do this it appears that pulling its reset pin low should solve the proble. Indeed, on one of the QLs modified it does isolate the master chip if the reset button is held down during switch on or off. Unfortunately it does not seem to give complete success on other machines. Obviously Sinclair knew more development was required! As the addition of the battery and diodes is relatively easy it seems worth-while trying it if you like this feature - there seems a good chance it may work on your QL.

To install the components:

(1.) Connect two AAA type alkaline cells in series. I soldered wires to mine but beware if you do this as alkaline cells MAY EXPLODE if overheated. (A suitable plastic battery clip is available from Radio Shack.--Ed.)

(2.) Remove the large chip marked ZX8302, IC 23 on the circuit board and bend out pin 40, the top RH pin. Take the usual precautions against static damage. While the chip is out, solder the cathode ends (the ends marked with coloured bands) of the two germanium diodes to the bent out pin. Also solder a wire (black for negative) to pin 20 (bottom left) of the chip. Sleeve the diodes and solder the loose end of one to pin 11 of the chip. Do not let solder get down the pins as all but pin 40 have to be reinserted in the holder. The master chip usually has two resistors soldered to its pins and laying on top of the IC (not always there--Ed.).

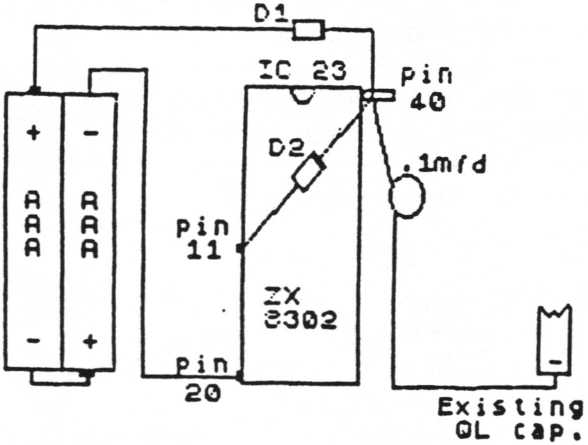
(3.) Solder one end of a small capacitor, 0.1 mfd or so, to a suitable groundpoint so its other end will reach pin 40 of the IC when it is replaced. I used the negative end of the small 100uf cap to the right of the master chip as a suitable ground.

(4.) Replace the IC

(5.) Fit the remaining loose diode wire to battery Positive and the black wire from pin 20 of the IC to battery Negative. Fit the loose end oof the added capacitor wire to Pin 40 where the two diode cathodes have been connected.

That's it. The battery should be insulated. I used electrician's tape. The battery may be positioned immediately to the left of the master chip on top of the two smaller ICs. I used a double-sided sticky tape on top of these ICs to hold the batteries firmly. Don't forget to sleeve the diodes. (heat-shrink tubing would probably work well--Ed.)

If you are lucky, your QL clock will maintain the correct time until the batteries fail in about a year. If you are not so lucky, you may occasionally need to reset the clock. If the clock is very unreliable you may wish to purchase more ZX8302s. (Not at $30 a pop on this side of the pond. I would just forget it. Norm Lehfeldt of SLUG, who originally reprinted this article, reports his clock works perfectly.--Ed)



```
   1 REM Copy a block of the scr
een...by John Pazmino. Reprinted
 from the May/Jume issue of LIST
, the newsletter of the Long Isl
and Sinclair Timex User group.

...Occasionally you'll want to c
opy to paper only a boxed-off se
ction of the screen as by doing 
COPY.  The demo here fOr the 100
0 does such a copy. ***For 2068 
line 1050 is LPRINT SCREEN$ (R,C
); OMIT LINE 1040.
   2 FOR I=0 TO 511
   4 PRINT CHR$ (RND*64)
   6 NEXT I
   8 COPY
  10 PRINT AT 21,0;" INPUT TOP R
OW"
  12 INPUT A
  14 PRINT AT 21,0;" THEN INPUT
BOTTOM ROW:                    "
  16 INPUT B
  18 PRINT AT 21,0;" THEN INPUT
LEFT COLUMN                    "
  20 INPUT X
  22 PRINT AT 21,0;" THEN INPUT
RIGHT COLUMN                   "
  24 INPUT Y
  26 LPRINT "BLOCK FOR ROWS ";A;
" TO ";B;" AND ","COLUMNS";X;" T
O ";Y;":      ",,,
  28 GO SUB 1000
  30 STOP
1000 REM LPRINT A BLOCK OF SCREE
N
1010 REM LINE 1010 FAST ON 1000
1020 FOR R=A TO B
1030 FOR C=X TO Y
1040 PRINT AT R,C
1050 LPRINT CHR$ PEEK (PEEK 1639
8+256*PEEK 16399);
1060 NEXT C
1070 LPRINT
1080 NEXT R
1090 REM LINE 1090 SLOW ON 1000
1100 RETURN
```

BUILD A SPECTRUM ROMSWITCHING CIRCUIT FOR YOUR TS-2068

By Gary Lessenberry

Reprinted from the May issue of the N/L of the Chicago Area T/S Users Group

I recently purchased some Spectrum ROMs from Zebra Systems with the hope that I might be able to make my own Spectrum romswitching circuit without paying the high price of those circuits that are commercially available. When I examined the TS-2068's ROM circuitry, I realized that this was an easier task than I had originally assumed! All that I needed was: a Spectrum ROM, an SPDT toggle switch (Radio Shack #275-6725), two feet of insulated wire and two 10K ohm resistors (Radio Shack #271-133)

To start the project, you first remove the top from your computer case by removing the seven screws in the bottom of the case. When you look inside, it will appear as in figure 1. You now remove the Timex ROM (U16). To remove it, gently pry it with a small screwdriver or knife inserted between the socket and the ROM.

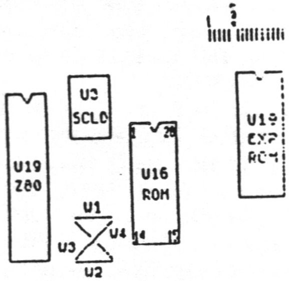


FIGURE A

It is important that you test your Spectrum ROM before constructing this circuit. To test it, place it in the socket from which you have removed the TS-2068 ROM and energize your computer. The Sinclair copyright should be displayed. If not, your ROM may be defective. After the test, remove the Spectrum ROM.

Take your Timex ROM and place your Spectrum ROM directly over it with the notches in the same direction (see figure 3). There should only be a thin space between the two ROMs and all of their leads should be touching. Do not leave a lot of space between these ROMs because clearance is critical when you reassemble you computer! You will now, very gently, bend pin 20 on both ROMs upward untill they are perpindicular to the other pins. You may now solder all of the pins except pin 20. Be careful when soldering. Allow 30 seconds between the soldering of each pin so that you won't overheat and damage the ROMs. To pin 20 of each ROM, you will solder a peice of wire and one end of a 10K ohm resistor. The other end of each 10K ohm resistor will be soldered to pin 28 (+5vdc). The other end of the two wires that you have comming from pin 20 of the two ROMs will be soldered to the toggle switch. The toggle switch has three pins on it. Two of these pins are labled "ON". Solder one wire to each of these two pins. Another wire will be soldered to the middle pin of the the toggle switch with the other end of that wire going to the circuit board and soldered to V1.

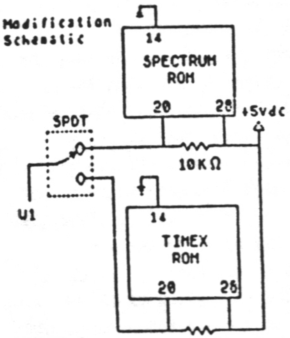
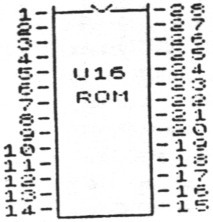
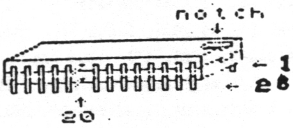


FIGURE 2



ROM Pin Assignments



Spectrum ROM Piggybacked On A Timex ROM

FIGURE 3

DUNGEON OF YMIR (TS1500)

Brief Description: Dungeon of Ymir is a multi-level maze adventure game written completely in machine code. Ymir requires 24K of RAM, and versions are available for the 1500 with a Hunter Board or other 8-16K RAM, OR for the 32K 1500 (with 16k Rampack). Ymir is true HI-RES, using a HI-RES technique which works only with the 1500 (NOT the 1000 or ZX81). A version for the ZX81/TS1000 is scheduled for release in the near future.

Dungeon of Ymir is available for $24.95 from FRED NACHBAUR, C-12, MTN. STN. GROUP BOX, NELSON, BC, CANADA V1L 5P1. Fred also has many other fine hardware and software products for the ZX81 family. Write for details.

WHAT IS THIS YMIR??

Upon loading (in 2 parts), a 64 column cover screen displays the Cast of Characters, and control keys. The object of the game is to successfully make it to the 9th level of the maze, retrieve the SWORD OF KASLO, and return it to your village. The game operates in the traditional form of the genre: you have a certain number of Hit Points (damage you can sustain), based on your Experience level, healing rate, and on what spells and potions you have at your disposal. On your journey, you will encounter 16 different types of monsters, lamps, mystery chests (which can be either very good or very bad), pits, ceiling holes, stairs up and down, gold (which is "cashed in" for Experience Points at the Temple of Asylum on each level), and an Oracle, who can hand out useful gifts, make suggestions, and punish impudence.

There are also several types of Spell Vials, which may be simply found in the maze, or in a mystery box, or handed out by the Oracle. These include Teleport spells, Rejuvenation spells, Healing potions, Drift spells, Shield spells, etc.

The playing screen is very impressive, with hi-res UDGs for all the monsters and objects. The maze is revealed only as you travel through it. As you proceed to lower levels, monsters become more difficult to cope with. Monsters are in constant motion, which starts randomly and takes on more evil intent as the game progresses! Critical information is displayed throughout the game on the bottom line of the screen. A FAST-SAVE with autoboot reloading permits the game to be saved in progress at any time, and reloaded in 70 seconds. I have found the fast save/load to be very reliable.

Conclusions: I must confess that I was very skeptical when I first loaded YMIR, since I've never been a D&D-type game afficionado. ("OK, I'll load Fred's new game, just to be polite...") My skepticism quickly turned to interest, curiosity, frustration, fascination, addiction. The graphics and animation are excellent (the little man bends over when you pick something up!), and the presentation of auxiliary screens (progress reports, etc.) is snappy and eye-catching. Dungeon of Ymir is, in the words of its author, "easy to play, difficult to master". I'm hooked, and I have yet to live past the 3rd level. This game has even made me late for work. Dungeon of Ymir is five-star software: a MUST-HAVE.
You've really done it this time, Fred!

Communications

Well, I finally got my hands on a more-or-less professional BBS communications package and no, it's not Tinyboard by Flashware. It's a system called Casboard 2068 (the Flexi-Board System) by Kurt A. Casby. It is comprised of a block of code that handles the Xmodem protocol for uploads and downloads and a block of basic for the normal menus and the like. It comes on a tape with versions for tape(with no upload/download), microdrive and disk. To all intents and purposes, it appears to the caller as a professional system with multiple message bases, upload and downloads, chat with the sysop(if he's there that is , and an extensive set up routine possibility for the sysop. It even requires a common password arrangement so that only those to whom you have provided the same, can use the BBS.

I had a number of occasions to test the BBS system with a number of my friends and all except for the auto save to disk of an uploaded program the BBS operated as advertised; ie perfectly. The problem of the upload to the disk appears to be more of a problem with the caller's software (Specterm-64) rather than the BBS program, but that has as yet to be proven.

There is a standard setup routine that the sysop must follow in order for the program to function correctly. If this process is not followed to a 't', then the system will stop with an error that is usually good enough to stop the program in it's tracks. The sysop must identify the password, upload and download permissability, message base usage and a few other misc items. After doing this for the first time and subsequently saving to disk, future setups can be quite quick. The sysop then sits back and watches TV, or to some poor slob routing around through the BBS. You can always force a chat routine with the user if the routing becomes crazy.

I had thought about setting up the system on a special time and date routine for my local club and had thought that the best hours would be between 8 and 10 p.m. on a Saterday and Sunday evening, particularily because of the phone ringing, or so I thought. What I subsequently found out was that the BBS program answers the phone so fast that the telephone doesn't have a chance to ring. This is great to one point of view, but it was particularily disconcerting to my mother-in-law to receive the rather harsh tones of a high frequency modem rather than the sweet melodic tones of my voice. She called me 2 hours later with a few words to the wise.

In any case, I am, through all this rambling, recommending this Casboard BBS system to all who would like to try their hands at being a sysop. Great fun and a sense of accomplishment!
Roelof Mulder

queens dominant

This program lets you distribute chess queens on a board, and shows which squares they are attacking. The aim is to attack all of the board with as few queens as possible, by judicious choice. It *can* be done with five of them.

```
10 DIM B (8, 8)
20 FOR I = 1 TO 8
30 FOR J = 1 TO 8
40 PRINT AT 2 * I, 2 * J + 8; "gH"
50 NEXT J
60 NEXT I
70 LET P = 2
80 LET Q = 9
90 GOSUB 2000
92 LET P0 = P
94 LET Q0 = Q
100 IF INKEY$ < > "" THEN GOTO 100
110 IF INKEY$ = "" THEN GOTO 110
120 LET I$ = INKEY$
130 IF I$ = "Q" THEN GOTO 200
140 IF I$ = "5" THEN LET Q = Q - 2
150 IF I$ = "6" THEN LET P = P + 2
160 IF I$ = "7" THEN LET P = P - 2
170 IF I$ = "8" THEN LET Q = Q + 2
180 GOSUB 1000
185 GOSUB 1020
190 GOTO 92
200 PRINT AT P, Q + 1; " Q "
207 LET J = (Q - 7) / 2
210 LET B (I, J) = 1
220 FOR T = 1 TO 8
230 IF B (I, T) = 0 THEN LET B (I, T) = -1
240 IF B (T, J) = 0 THEN LET B (T, J) = -1
250 NEXT T
260 FOR F = -1 TO 1 STEP 2
270 FOR T = -8 TO 8
280 LET U = I + F * T
290 LET V = J + T
300 IF U < 1 OR U > 8 OR V < 1 OR V > 8 THEN GOTO 320
310 IF B (U, V) = 0 THEN LET B (U, V) = -1
320 NEXT T
330 NEXT F
340 GOSUB 2000
350 GOTO 92
1000 PRINT AT P, Q; " > "
1010 RETURN
1020 PRINT AT P0, Q0; "□"
1030 RETURN
2000 FOR I = 1 TO 8
2010 FOR J = 1 TO 8
2020 IF B (I, J) = -1 THEN PRINT AT 2 * I, 2 * J + 8; "+"
2030 NEXT J
2040 NEXT I
2050 RETURN
```

★★★ SUMMER MEGA-ISSUE ★★★

VSUG

The Vancouver Sinclair Users Group has been in existence sin 1982. We are a support group for the owners and users of SINCLAIR and TIMEX computers.

Pres:- Ken Abramson PH. # (604) 438-7740
V/Pres:- 'Rusty' Townsend
Sec:- Harvey Taylor
Treas. & Editor:- Rod Humphreys

Our membership dues are only $15.00/year and may be sent to t Treasurer:

Rod Humphreys
2006 Highview Place
Port Moody, B.C., V3H 1N5

Members of VSUG receive a monthly issue of ZXAppeal - o newsletter.

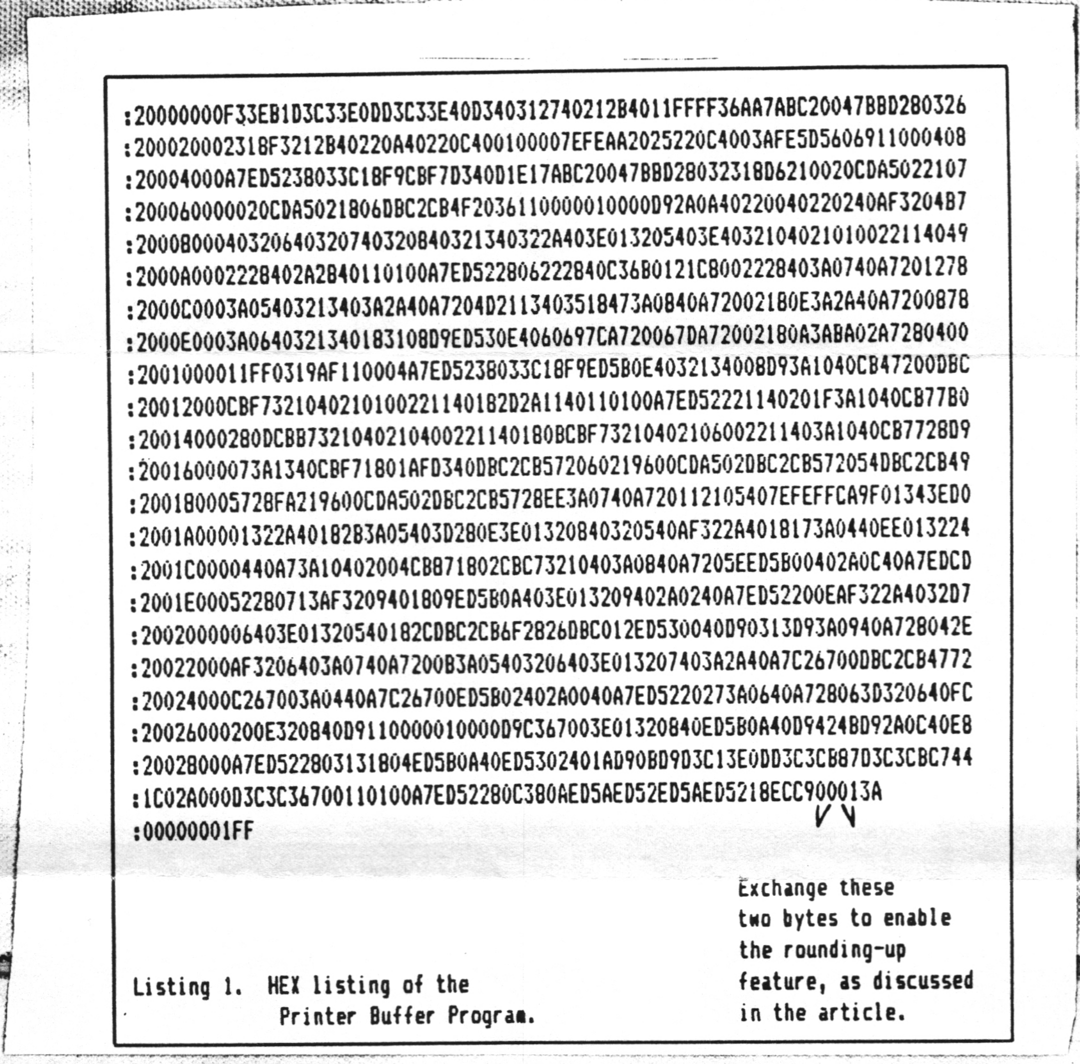
ZXAppeal accepts advertising. Our **PREPAID** rates are:

$20.00 -- full page
$12.00 -- 1/2 page
$8.00 -- 1/4 page

ZXAppeal has a print run of 75 copies per month for members and distributed to approx 40 other SINCLAIR User Groups througho North America as well as overseas via the NETWORK.

NETWORK correspondence may be directed to the Editor at the abo address.

All articles appearing in ZXAppeal may be reprinted by other T User Groups as long as credit is given to the author and VSUG.



```
:20000000F33EB1D3C33E0DD3C33E40D340312740212B4011FFFF36AA7ABC20047BBD280326
:200020002318F3212B40220A40220C400100007EFEAA2025220C4003AFE5D5606911000408
:20004000A7ED5238033C18F9CBF7D340D1E17ABC20047BBD28032318D6210020CDA5022107
:200060000020CDA5021806DBC2CB4F2036110000010000D92A0A40220040220240AF3204B7
:200080004032064032074032084032134032 2A403E013205403E4032104021010022114049
:2000A0002228402A2B40110100A7ED522806222B40C36B0121C8002228403A0740A7201278
:2000C0003A05403213403A2A40A7204D2113403518473A0840A72002180E3A2A40A7200878
:2000E0003A064032134018310 8D9ED530E4060697CA720067DA72002180A3ABA02A7280400
:2001000011FF0319AF110004A7ED5238033C18F9ED5B0E40321340 08D93A1040CB47200DBC
:20012000CBF7321040210100221140182D2A1140110100A7ED52221140201F3A1040CB77B0
:20014000280DCBB7321040210400221140180BCBF7321040210600221140 3A1040CB7728D9
:20016000073A1340CBF71801AFD340DBC2CB572060219600CDA502DBC2CB572054DBC2CB49
:200180005728FA219600CDA502DBC2CB5728EE3A0740A720112105407EFEFFCA9F01343ED0
:2001A00001322A40182B3A05403D280E3E01320840320540AF322A4018173A0440EE013224
:2001C0000440A73A10402004CB871802CBC73210403A0840A7205EED5B00402A0C40A7EDCD
:2001E000522807 13AF3209401809ED5B0A403E013209402A0240A7ED52200EAF322A4032D7
:2002000006403E01320540182CDBC2CB6F2826DBC012ED530040D90313D93A0940A728042E
:20022000AF3206403A0740A7200B3A054032064 03E013207403A2A40A7C26700DBC2CB4772
:20024000C267003A0440A7C26700ED5B02402A0040A7ED5220273A0640A728063D320640FC
:20026000200E320840D9110000010000D9C367003E01320840ED5B0A40D9424BD92A0C40E8
:20028000A7ED522803131804ED5B0A40ED5302401AD90BD9D3C13E0DD3C3CB87D3C3CBC744
:1C02A000D3C3C36700110100A7ED52280C380AED5AED52ED5AED5218ECC900013A
:00000001FF
```

Exchange these
two bytes to enable
the rounding-up
feature, as discussed
in the article.

Listing 1. HEX listing of the
Printer Buffer Program.

GOES WITH
PAGE 25
SUMMER '87